LabVIEW™

Version 5.1 Addendum

NATIONAL INSTRUMENTS™

February 1999 Edition
Part Number 322179A-01

Internet Support
E-mail: support@natinst.com
FTP Site: ftp.natinst.com
Web Address: http://www.natinst.com

Bulletin Board Support
BBS United States: 512 794 5422
BBS United Kingdom: 01635 551422
BBS France: 01 48 65 15 59

Fax-on-Demand Support
512 418 1111

Telephone Support (USA)
Tel: 512 795 8248
Fax: 512 794 5678

International Offices
Australia 03 9879 5166, Austria 0662 45 79 90 0, Belgium 02 757 00 20, Brazil 011 288 3336,
Canada (Ontario) 905 785 0085, Canada (Québec) 514 694 8521, Denmark 45 76 26 00, Finland 09 725 725 11,
France 01 48 14 24 24, Germany 089 741 31 30, Hong Kong 2645 3186, Israel 03 6120092, Italy 02 413091,
Japan 03 5472 2970, Korea 02 596 7456, Mexico 5 520 2635, Netherlands 0348 433466, Norway 32 84 84 00,
Singapore 2265886, Spain 91 640 0085, Sweden 08 730 49 70, Switzerland 056 200 51 51, Taiwan 02 377 1200,
United Kingdom 01635 523545

National Instruments Corporate Headquarters
6504 Bridge Point Parkway Austin, Texas 78730-5039 USA Tel: 512 794 0100

© Copyright 1999 National Instruments Corporation. All rights reserved.

Important Information

Warranty

The media on which you receive National Instruments software are warranted not to fail to execute programming instructions, due to defects in materials and workmanship, for a period of 90 days from date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace software media that do not execute programming instructions if National Instruments receives notice of such defects during the warranty period. National Instruments does not warrant that the operation of the software shall be uninterrupted or error free.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. National Instruments will pay the shipping costs of returning to the owner parts which are covered by warranty.

National Instruments believes that the information in this manual is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, National Instruments reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult National Instruments if errors are suspected. In no event shall National Instruments be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, NATIONAL INSTRUMENTS MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF NATIONAL INSTRUMENTS SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. NATIONAL INSTRUMENTS WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF. This limitation of the liability of National Instruments will apply regardless of the form of action, whether in contract or tort, including negligence. Any action against National Instruments must be brought within one year after the cause of action accrues. National Instruments shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failures caused by owner's failure to follow the National Instruments installation, operation, or maintenance instructions; owner's modification of the product; owner's abuse, misuse, or negligent acts; and power failure or surges, fire, flood, accident, actions of third parties, or other events outside reasonable control.

Copyright

Under the copyright laws, this publication may not be reproduced or transmitted in any form, electronic or mechanical, including photocopying, recording, storing in an information retrieval system, or translating, in whole or in part, without the prior written consent of National Instruments Corporation.

Trademarks

ComponentWorks™, CVI™, DataSocket™, HiQ™, LabVIEW™, `natinst.com`™, National Instruments™, NI-488.2™, NI-DAQ™, and NI-VXI™ are trademarks of National Instruments Corporation.

Product and company names listed are trademarks or trade names of their respective companies.

WARNING REGARDING MEDICAL AND CLINICAL USE OF NATIONAL INSTRUMENTS PRODUCTS

National Instruments products are not designed with components and testing intended to ensure a level of reliability suitable for use in treatment and diagnosis of humans. Applications of National Instruments products involving medical or clinical treatment can create a potential for accidental injury caused by product failure, or by errors on the part of the user or application designer. Any use or application of National Instruments products for or involving medical or clinical treatment must be performed by properly trained and qualified medical personnel, and all traditional medical safeguards, equipment, and procedures that are appropriate in the particular situation to prevent serious injury or death should always continue to be used when National Instruments products are being used. National Instruments products are NOT intended to be a substitute for any form of established process, procedure, or equipment used to monitor or safeguard human health and safety in medical or clinical treatment.

Contents

About This Addendum

Chapter 1
Required Configuration, Installation, and Upgrade Information

Required System Configuration....1-1
Operating System Patches....1-4
Concurrent PowerMAX....1-4
Distribution Changes in LabVIEW 5.1....1-5
LabVIEW 5.1 Platforms....1-5
Discontinued Media....1-5
Installing LabVIEW....1-6
LabVIEW RT....1-6
Windows....1-6
Macintosh....1-7
UNIX....1-7
For More Information about LabVIEW....1-10
Data Acquisition, VXI, and GPIB Installation Notes....1-10
Installing HiQ for Windows....1-12
Examples and Solutions for Your LabVIEW Programs....1-12
Low-Level Register I/O for Windows 95/98....1-13
Common LabVIEW Launch Errors on UNIX....1-13
Configuring LabVIEW Windows on UNIX....1-14
Configuring LabVIEW with the Tab Window Manager....1-14
Configuring LabVIEW with the HP VUE Window Manager....1-14
Configuring LabVIEW with the Motif or CDE Window Manager....1-14
Notice to Sun SPARCstation 5 Users....1-15
Compatibility Issues between Versions 4.1 and 5.*x*....1-16
Compatibility VIs for New Server Functionality....1-16
Compatibility VIs for ActiveX Functions....1-16
Additional Resources....1-17
Upgrading to LabVIEW 5.1....1-17
Converting VIs....1-18
Upgrading Application Libraries and Toolkits....1-18

Chapter 2
New Features in LabVIEW 5.1

Dialog Box, Menu, and Window Enhancements 2-1
Scaling Front Panel Objects 2-1
Saving for a Previous Version 2-4
Accessing Recently Opened Files 2-4
Searching in LabVIEW 2-5
Macintosh Navigation Services in LabVIEW 2-6
New VIs, Functions, and Controls 2-7
Changes to Controls and Indicators 2-7
3D Graph Controls for Windows 2-9
Enhancements to Property and Invoke Nodes 2-9
Improvements to VI Server Properties in Reserved VIs and Runtime Systems 2-9
Ring Enhancements 2-11
ActiveX Enhancements for Windows 2-13
ActiveX Event Functions for Windows 2-14
HiQ and MATLAB Functionality for Windows 2-15
Integration of Mathematics and Signal Processing VIs 2-22
Integration of the Picture Control VIs 2-24
Sound VIs for Windows and Macintosh 2-25
Generating Reports in LabVIEW for Windows 2-25
Report Generation VI Descriptions 2-28
Enhancements to DAQ 2-39
DAQ Solution Wizard 2-39
Support for NI-DAQ for Windows and Macintosh 2-39
New Syntax Element for Nonsequentially Scanned SCXI Module Channels 2-39
Enhancements to VISA 2-40
Enhancements for Building and Distributing Applications 2-41
Building Executable Programs 2-41
Run-Time Engine for the Application Builder for Windows 2-46
Enhancements to Networking 2-47
DataSocket VIs for Windows 2-47
Internet/HTTP Services 2-47
Enhancements to Examples and Activities 2-62

Appendix A
Manual Clarifications and Additions

Multithreading A-1
ActiveX A-1
Instrumentation A-2
General Interface Features A-2
Icon and Text Palettes A-2
File Manager Tool A-3
Other General Interface Features A-5
Support for Template VIs and Controls A-7
Adding VIs to the Project and Help Menus A-7
Allocation of Threads on Concurrent PowerMAX and Solaris 2 ..... A-7
Clarifications to the LabVIEW User Manual A-8
VISA Error Codes A-9

Index

Figures

Figure 2-1. Dialog Palette 2-8
Figure 2-2. Ring Constant Examples 2-11
Figure 2-3. Setting a Data Value with a Numeric Constant 2-12
Figure 2-4. Setting a Data Value with a Ring Constant 2-12
Figure 2-5. ActiveX Events Palette 2-14
Figure 2-6. Web Server Browser Access Dialog Box 2-49
Figure 2-7. Web Server Visible VIs Dialog Box 2-52

Figure A-1. File Manager Tool Dialog Box A-4

Tables

Table 1-1. Installation Requirements 1-2
Table 1-2. Common LabVIEW Launch Errors on UNIX 1-13

Table 2-1. HiQ and MATLAB Datatypes in LabVIEW 2-21
Table 2-2. Script Node Runtime Error Messages 2-22
Table 2-3. Mathematics VIs Current and Previous Locations 2-23
Table 2-4. Signal Processing VIs Current and Previous Locations 2-24
Table 2-5. Token Descriptions 2-27
Table 2-6. Web Server Configuration Dialog Box Options 2-48
Table 2-7. Web Server Browser Access Dialog Box Options 2-50
Table 2-8. Examples of Access List Entries 2-51
Table 2-9. Web Server Visible VIs Dialog Box Options 2-53
Table 2-10. Wildcard Characters in Visible VIs List 2-54
Table 2-11. Examples of Visible VI List Entries 2-54

About This Addendum

This addendum describes all of the new LabVIEW 5.1 features, compatibility issues, and configuration, installation, and upgrade information. Upgrade issues pertain to LabVIEW 5.1 only.

Note ***LabVIEW 5.1 ships with the same manual set that shipped with LabVIEW 5.0. This addendum updates that documentation. However, enhancements to LabVIEW 5.1 have rendered some information in the manuals incorrect. Please refer to Appendix A, Manual Clarifications and Additions, for corrections and important information.***

This document is divided into the following sections:

- Chapter 1, *Required Configuration, Installation, and Upgrade Information*, describes the system requirements for the LabVIEW software and contains installation instructions and updated documentation information.
- Chapter 2, *New Features in LabVIEW 5.1*, describes the features added and the significant changes made between versions 5.0 and 5.1. To help you learn more about LabVIEW, version 5.1 offers extensive online documentation, which you can access by choosing **Help**»**Online Reference...**.
- Appendix A, *Manual Clarifications and Additions*, clarifies and corrects information in the LabVIEW manual set.
- The *Index* contains an alphabetical list of key words and topics in this manual, including the page where you can find each one.

Regardless of whether you are a new or upgrade user, begin by reading Chapter 1, *Required Configuration, Installation, and Upgrade Information* before continuing with this installation.

If you are upgrading from a previous version of LabVIEW, carefully read the *Upgrading to LabVIEW 5.1* section in Chapter 1, *Required Configuration, Installation, and Upgrade Information*.

Note ***LabVIEW is Year-2000 compliant. Because LabVIEW has never stored two-digit years, the change to 2000 does not affect any internal storage of dates.***

1

Required Configuration, Installation, and Upgrade Information

This chapter describes the system requirements for the LabVIEW software and contains installation instructions and updated documentation information.

Read the *Required System Configuration* section, then follow the instructions in the *Installing LabVIEW* section of this chapter.

Required System Configuration

Table 1-1 describes the minimum system requirements needed to run LabVIEW 5.1.

Table 1-1. Installation Requirements

Platform	Media and System Requirements	Important Notes
All Windows Versions	Distributed on CD-ROM. A separate CD contains the complete instrument driver library. You need a minimum of 60 MB of disk storage space for the Base package or 85 MB for the Full Development System. You need a minimum of 16 MB of RAM.	The *LabVIEW Online Tutorial* default configuration requires the LabVIEW 5.1 distribution CD to be in your CD-ROM drive. You also can install the *LabVIEW Online Tutorial* files on your hard drive. This installation requires approximately 40 MB of hard disk space. The *LabVIEW Online Tutorial* and LabVIEW Help files contain 256-color graphics. Your video driver, configured through **Control Panels»Display**, must be configured for at least 256 colors. Minimum requirements to view the tutorial are 800 × 600 pixel resolution and the Microsoft Video for Windows driver. To view Help files, configure your video driver for at least 256 colors with 800 × 600 pixel resolution. National Instruments recommends 32 MB of RAM and a Pentium processor for this version to run effectively.
Windows 95/98	LabVIEW runs on any system that supports Windows 95/98.	
Windows NT	LabVIEW runs on Windows NT 4.0 Service Pack 3 or later. DEC Alpha, MIPS, and PowerPC 80*x*86 emulators must emulate 80386 instructions to run LabVIEW.	

Table 1-1. Installation Requirements (Continued)

Platform	Media and System Requirements	Important Notes
Power Macintosh	Distributed on CD. LabVIEW requires System 7 or 8. You need a minimum of 24 MB of RAM and at least 100 MB of disk storage space for the minimal installation of LabVIEW or 120 MB for the full installation.	National Instruments recommends that you have at least 32 MB of RAM. You might need more memory, depending on the size of the application you design in LabVIEW and the amount of data that your application manipulates. For more accurate timing, install the Apple QuickTime extension. When you use QuickTime, timing accuracy should increase from 16.6 ms resolution to approximately 1 ms resolution. System response varies depending on background applications, other extensions, networking activity, and disk caching.
All UNIX Versions	LabVIEW requires an X Window System server, such as OpenWindows 3.*x*, HP-VUE, or X11R6. You need a minimum of 32 MB of RAM with 32 MB of swap space storage. You need a minimum of 65 MB of disk storage space for the entire LabVIEW package.	LabVIEW uses a directory for storing temporary files. Some of the temporary files are large, so keep several megabytes of disk space available for this temporary directory. The default for the temporary directory is `/tmp`. You can change the temporary directory by selecting **Edit»Preferences...**. If LabVIEW aborts unexpectedly, it might leave files behind in the temporary directory. Remove old files occasionally to avoid depleting your disk space. To save space, install only the VIs you plan to use. LabVIEW does not require a specific graphical user interface (GUI) such as Motif or OpenLook, because LabVIEW uses `Xlib` to create its own GUI.
Sun	Distributed on CD. LabVIEW runs on SPARCstations with Solaris 2.4 or later.	

Table 1-1. Installation Requirements (Continued)

Platform	Media and System Requirements	Important Notes
HP-UX	Distributed on CD. LabVIEW runs on Hewlett-Packard Model 9000 Series 700 computers with HP-UX 10.20 or later.	HP workstations limit the size of a process such as LabVIEW to 64 MB. You may need to increase this setting to accommodate your LabVIEW application. For information on changing this setting, see the *HP-UX 10.x* section under *Installing LabVIEW*.
Linux	Distributed on CD. LabVIEW runs on Linux for Intel x86 processors, version 2.0.x or later.	Requires GNU C Library Version 2 (`glibc2`, also known as `libc.so.6`). RedHat Linux 5.0 or later includes the `glibc2` runtime library.
Concurrent PowerMAX	Distributed on 4 mm DAT tape. LabVIEW runs on PowerMAX version 4.2 or later.	See the *Operating System Patches* section for Concurrent PowerMAX below for information on the patches you must download to run LabVIEW.

Operating System Patches

For your LabVIEW package to run effectively, be sure to install the following patches.

Concurrent PowerMAX

LabVIEW 5.1 requires version 4.2 or later of the PowerMAX operating system. In addition, you must install the following patches, available from Concurrent, so that basic LabVIEW networking functions correctly:

- inet-005
- One of the following, depending on your system:
  - base-007 for Power Hawk 610, Power Hawk 620, and all single-processor PowerStack systems
  - base-008 for Power Hawk 640 and multi-processor PowerStack II systems
  - base-009 for Night Hawk systems

Distribution Changes in LabVIEW 5.1

This section explains changes in platform support and installation media with this and subsequent versions of LabVIEW.

LabVIEW 5.1 Platforms

LabVIEW 5.1 is now available with limited support on Linux/*x*86. For more information, see `http://www.natinst.com/linux`.

LabVIEW 5.1 and planned future versions of LabVIEW are not available on the following platforms:

- Windows 3.*x*
- Macintosh for the Motorola 680*x*0
- Solaris 1.*x*
- HP-UX 9.*x*

You can purchase LabVIEW 5.0.*x* for these platforms and maintenance upgrades and fixes are available from National Instruments, but National Instruments plans no new feature development for these platforms. Contact National Instruments to purchase LabVIEW 5.0.*x*.

Note ***LabVIEW 5.1 has a* Save for Previous *option that can ease this transition. Notice that choosing this option only maintains window proportions. To have the objects maintain proportions, you must also choose* Scale All Objects on Panel. *Alternatively, you could choose a single object to scale when the window changes size. See the Saving for a Previous Version section of Chapter 2, New Features in LabVIEW 5.1, for more information.***

Discontinued Media

National Instruments ships LabVIEW 5.1 on CD-ROM only. If you do not have a CD-ROM drive on your system, contact National Instruments.

Installing LabVIEW

If you are upgrading from an earlier version of LabVIEW, read the *Upgrading to LabVIEW 5.1* section later in this chapter before installing LabVIEW.

(Windows) For an introduction to the LabVIEW environment, complete the *LabVIEW Online Tutorial*. Launch the tutorial by clicking **LabVIEW Tutorial** in the **LabVIEW** dialog box.

To access the LabVIEW dialog box, either launch LabVIEW or close all open LabVIEW VIs if you already are running LabVIEW.

(Windows) When you insert the LabVIEW installation CD, a dialog box appears, from which you can choose to install LabVIEW, install HiQ, or run the toolkit demos.

LabVIEW RT

LabVIEW RT works with National Instruments RT Series intelligent DAQ hardware, and allows you to perform real-time, deterministic data acquisition and to control applications on Windows PCs. With LabVIEW RT, you can create embedded VIs that run in real-time on processor-based RT Series hardware. Combined with the RT Series intelligent DAQ boards, LabVIEW RT gives you the simple graphical programming of LabVIEW with proven DAQ technology for a complete, integrated solution for real-time applications.

If you are installing LabVIEW RT instead of LabVIEW 5.1, refer to the *RT Series Hardware and LabVIEW RT User Manual* for complete installation instructions.

Caution ***LabVIEW RT is English only. If you are upgrading from a non-English version of LabVIEW to LabVIEW RT, your version of LabVIEW will be in English.***

Windows

Complete the following steps to install LabVIEW for Windows.

1. **(Windows NT)** Log on to Windows NT as an administrator or as a user with administrator privileges.
2. If you are installing LabVIEW for Windows 95/98/NT, run *x*:\AUTORUN.EXE, where *x* is the drive letter for your CD-ROM drive.

Note ***The installer gives you the option of performing a full installation or a minimal installation. If you do not have sufficient disk space (approximately 85 MB), choose the minimal installation and use your LabVIEW CD to access the remaining components.***

3. After you choose an installation, follow the instructions that appear on your screen.
4. After you have completed the installation, LabVIEW is ready to run. If you plan to use DAQ or GPIB devices with LabVIEW, you must restart your computer to load the new drivers.

If you have installed LabVIEW on a server, new users might want to copy the `Activity` directory from the server to their local machine. You use the `Activity` directory to complete activities that illustrate basic LabVIEW concepts. You can find these activities in the *LabVIEW User Manual* and the LabVIEW *Online Reference*, which you can access by selecting **Help»Online Reference…**.

Macintosh

Complete the following steps to install LabVIEW for Macintosh.

1. Insert the LabVIEW installation CD into your computer's CD-ROM drive.
2. Run the `LV51 PMAC Installer`. The installer defaults to the Easy Install mode, which installs all LabVIEW examples, DAQ VIs, and NI-DAQ 6.1 with drivers. You can choose to install each component separately by choosing **Custom Install** from the drop-down menu.
3. Follow the instructions that appear on your screen.

UNIX

Complete the following steps to install LabVIEW for UNIX.

Solaris 2

1. To enable superuser privileges, type `su root` and enter the root password.
2. Insert the LabVIEW CD. On Solaris 2.*x*, the CD automatically mounts as soon as the CD is inserted into the drive. If this feature is disabled on your workstation, you must mount the CD by typing the following command:

   ```
   mount -o ro -F hsfs /dev/dsk/c0t6d0s2 /cdrom
   ```

3. If your CD was mounted automatically, type the following command:

   ```
   pkgadd -d /cdrom/cdrom0/solaris2
   ```

4. If you used the mount command in step 2, type the following command:

   ```
   pkgadd -d /cdrom/solaris2
   ```

☞ **Note** ***See the*** `README` ***file in*** `/cdrom/cdrom0/solaris2` ***or*** `/cdrom/solaris2` ***for instructions on custom installation or other additional information.***

5. Follow the instructions on your screen.

HP-UX 10.*x*

By default, HP workstations limit the size of a process such as LabVIEW to 64 MB. You can change this setting by adjusting a kernel configuration parameter that limits the amount of data a process can use. To edit this parameter, enable superuser privileges by typing `su root` and entering the root password. Use the SAM system administration utility to view the list of kernel configuration parameters. From SAM, go to **Kernel Configuration»Configurable Parameters** and change the value of the **maxdsiz**, or **Maximum Data Segment Size (bytes)** parameter to a larger value. If you need to rebuild the kernel and reboot for changes to take effect, the SAM utility guides you through this process.

1. To enable superuser privileges, type `su root` and enter the root password.
2. Mount the LabVIEW CD on the `/cdrom` directory with the SAM system administration utility.
3. To change to the installation directory, type the following command:

   ```
   cd /cdrom/HP-UX
   ```

4. To run the installation script, type the following command:

   ```
   ./INSTALL
   ```

5. Follow the instructions on your screen.

Linux

To install LabVIEW 5.1 for Linux/x86, perform the following steps.

1. Login to your system as `root`.
2. Mount the CD-ROM.

3. To change the current directory to the mounted CD-ROM, type the following command:

   `cd /mnt/cdrom.`

4. To run the installation script, type the following command:

   `./INSTALL.`

   The `INSTALL` script prompts you to enter the directory where you want to install LabVIEW (typically `/usr/local` or `/opt`). The script uses RPM to install the RedHat package on RedHat Linux 5.0 or later systems, or it extracts the `.tar.gz` archives on other systems.

Note ***If your system does not have the required*** `glibc2` ***libraries installed, you have the option to install LabVIEW 4.1.1 instead. The file*** `Glibc2-HOWTO` ***on the CD gives detailed instructions on how to obtain and install*** `glibc2`**.**

You also can install the files by hand, using either RPM or glint on RedHat systems, or `tar/gunzip` on other systems. To install LabVIEW versions 4.1.1 and 5.1 simultaneously, use the `--force rpm` option.

Note ***Refer to the*** `README` ***file on the CD for additional installation instructions.***

PowerMAX

1. Insert the 4 mm DAT tape into the tape drive.
2. To create the directory in which you will install LabVIEW, type the following command:

   `mkdir /opt/lv51`

3. To change to the new directory, type the following command:

   `cd /opt/lv51`

4. Extract the files from the tape by typing the following command:

   `tar xv`

5. To run the installation script, type the following command:

   `./INSTALL`

6. Follow the instructions on your screen.

For More Information about LabVIEW

After you have installed LabVIEW completely, it is ready to run.

The *LabVIEW User Manual* and the LabVIEW *Online Reference*, which you can access by selecting **Help»Online Reference…**, provide activities that illustrate basic LabVIEW concepts. If you want to complete these activities, copy the `Activity` directory from the `LabVIEW` directory to your home directory.

The LabVIEW documentation set, including the *Code Interface Reference Manual* and the *VXI VI Reference Manual*, is available in Portable Document Format (PDF) on the LabVIEW CD in the `manuals` directory. You can copy this directory or selected PDF files to the `LabVIEW\manuals` directory on your hard drive. You must have Adobe Acrobat Reader 3.0 or later installed to view these files.

If you are upgrading from a previous version of LabVIEW, read the *Upgrading to LabVIEW 5.1* section, later in this chapter. If you have one of the add-on toolsets, consider installing those files at this time.

Data Acquisition, VXI, and GPIB Installation Notes

All National Instruments GPIB interfaces and DAQ devices come with the drivers and other software you need to use them. LabVIEW also comes with the drivers and other software you need to use National Instruments hardware. While the drivers included with LabVIEW are the same NI-488.2 and NI-DAQ drivers National Instruments includes with its GPIB and DAQ hardware, the version numbers might differ. Always use the driver with the higher version number. You can determine which version of NI-DAQ you are using with LabVIEW by running the Get Device Information VI.

Windows

When you install LabVIEW, the installer places the application and most of the related files in a directory you specify. The default name of this directory is `LABVIEW`. If you install DAQ or GPIB VIs, the installer places additional files, described in the following sections.

Use the National Instruments Measurement & Automation Explorer, which runs with LabVIEW, to configure your hardware. For information about how to configure your particular DAQ device, refer to the Measurement & Automation Explorer Help.

You can find further information about the NI-DAQ driver in the NI-DAQ Read Me File. To view this file, click the **Start** button and select **Programs»LabVIEW»NI-DAQ Read Me File**.

Macintosh

The LabVIEW installation program installs a control panel and various extensions in your system folder:

- For GPIB, LabVIEW installs the `NI-488 Config` control panel, which contains the driver code that communicates with your GPIB devices. LabVIEW also installs extensions that your GPIB hardware and software require.
- For DAQ, LabVIEW installs the `NI-DAQ` extension, which contains driver code that communicates with your DAQ devices, along with several other libraries and extensions that support NI-DAQ 6.1.
- The `NI-DMA/DSP` extension contains DSP and DMA drivers used by DAQ, GPIB, and DSP drivers.

Sun

While installing LabVIEW, you can choose the NI-488.2M drivers for the GPIB hardware you are using (one of the following: SB-GPIB-TNT, GPIB-ENET, or GPIB-SCSI-A). The installer then installs that driver for you.

If you have a GPIB-SCSI-A, follow the installation instructions in the documentation that came with your original GPIB-SCSI-A hardware and software kit, including the *Getting Started with Your GPIB-SCSI-A and the NI-488.2M Software for the Sun SPARCstation* manual.

☞ **Note** ***LabVIEW does not work with the GPIB-1014 series (VME) devices or the original GPIB-SCSI box. It does work with the newer GPIB-SCSI-A box.***

A VXI device driver must be installed on your system to perform VXIbus operations from LabVIEW. Install the device driver for Solaris 2.*x* before beginning development. To install the VXI device driver, refer to the *Getting Started with Your VXI/VME-PCI8022 and the NI-VXI Software for Solaris* manual.

☞ **Note** ***National Instruments periodically updates drivers for GPIB and VXI. If you add new GPIB or VXI hardware for use with LabVIEW, the included drivers might supersede those sent with LabVIEW. Compare the version numbers and use the driver with the higher number.***

Installing HiQ for Windows

The Windows CD includes HiQ, a mathematics application from National Instruments. If you install HiQ from the LabVIEW installation CD, a registration dialog box prompts you to enter a registration number. Use your LabVIEW registration number in the HiQ registration dialog box. The latest version of LabVIEW includes functionality that supports HiQ. If you would like to take advantage of this functionality, but do not yet have HiQ, be sure to install HiQ after you have installed LabVIEW. For more information about this functionality, see the *HiQ and MATLAB Functionality for Windows* section in Chapter 2, *New Features in LabVIEW 5.1*.

Examples and Solutions for Your LabVIEW Programs

(Windows and Macintosh) If you are using data acquisition (DAQ) or instrument I/O and want to find examples or generate solutions for your LabVIEW programs, launch the DAQ Solution Wizard by clicking **Solution Wizard** in the LabVIEW dialog box. For more information about the Solution Wizard, see Chapter 3, *Data Acquisition*, and Chapter 4, *Instrumentation*, of the *LabVIEW QuickStart Guide*.

(Windows) To find any other type of example, open the Search Examples Help file by clicking **Search Examples** in the LabVIEW dialog box.

The `examples` directory contains a VI named `readme.vi`. With this VI, you can find the available examples. When you select a VI, you can see the documentation that was entered for that VI by choosing **Window»Show VI Info...**. To open a VI, choose **File»Open...**.

☞ **Note** ***Because the controls and functions palettes changed with this release of LabVIEW, many paths to examples are listed incorrectly or not listed at all in the LabVIEW documentation set. Please see Chapter 2, New Features in LabVIEW 5.1, for more information on the updated activities and examples.***

Low-Level Register I/O for Windows 95/98

LabVIEW is very similar on all Windows operating systems. Unless your application communicates with hardware that is not supported by one of the operating systems, you can transfer VIs to other operating systems without any problems or having to make any modifications.

LabVIEW has two VIs named In Port and Out Port that you can use to read or write hardware registers. These VIs work under Windows 95/98. Windows NT applications, however, cannot manipulate hardware directly. If you need to communicate with a hardware device in Windows NT, you must write a Windows NT driver.

Common LabVIEW Launch Errors on UNIX

Table 1-2 lists common errors that might occur when you launch LabVIEW for UNIX. See the *Required System Configuration* section of this chapter for more information about solving these and other installation problems.

Table 1-2. Common LabVIEW Launch Errors on UNIX

Error Message/Description	Error Message/Description
`Xlib: connection to :0.0 refused by server`	**Probable Cause**—Trying to run LabVIEW as a user who does not have permission to open a window on the display server. Typically seen after running the `su` command to temporarily become a different user, such as root (superuser). **Solution**—Exit the `su` command and launch LabVIEW as the login user.
`Client is not authorized to connect to server`	
`Internal error during connection authorization check`	
`"Executable version doesn't match resource file"`	**Probable Cause**—Version of LabVIEW executable does not match version of `labview.rsc`. **Solution**—Verify that the `appResFilePath` parameter in the configuration file correctly sets the path to the `labview.rsc` file.

Configuring LabVIEW Windows on UNIX

This section describes procedures for configuring LabVIEW windows on UNIX operating systems.

Configuring LabVIEW with the Tab Window Manager

If you use the Tab Window Manager (`twm`), you can change environment settings so that `twm` interacts better with LabVIEW. Notice that with `twm`, you cannot close the floating palette menus in LabVIEW if these windows do not have title bars. To correct this problem, add the following line to your `.twmrc` file in your home directory:

```
DecorateTransients
```

This line adds title bars to the floating windows so you can close them.

Configuring LabVIEW with the HP VUE Window Manager

If you use the HP VUE Window Manager (`vuewm`), you can change environment settings so that `vuewm` interacts better with LabVIEW. By default, `vuewm` does not incorporate the window position requests of an application. Consequently, LabVIEW windows—such as the **Panel**, **Diagram**, **Help**, and **File** dialog windows—do not appear in consistent locations on your screen. To change the `vuewm` behavior, use the `xrdb` command to set two `vuewm` settings:

```
Vuewm.clientAutoPlace: False
Vuewm.positionIsFrame: False
```

To add the two entries, you also can edit the following files manually:

```
$HOME/.vue/sessions/home/vue.resources
$HOME/.vue/sessions/current/vue.resources
```

Configuring LabVIEW with the Motif or CDE Window Manager

If you use the Motif Window Manager (`mwm`) or the Common Desktop Environment (CDE) Window Manager (`dtwm`), you can change environment settings so that `mwm` or `dtwm` interacts better with LabVIEW. By default, `mwm` and `dtwm` do not incorporate the window position requests of an application. Consequently, LabVIEW windows—such as the **Panel**, **Diagram**, **Help**, and **File** dialog windows—do not appear in consistent locations on your screen.

(Motif) To change the behavior of `mwm`, use the `xrdb` command to set two `mwm` settings:

```
mwm.clientAutoPlace: False
mwm.positionIsFrame: False
```

(CDE) To change the behavior of `dtwm`, use the `xrdb` command to set two `dtwm` settings:

```
dtwm.clientAutoPlace: False
dtwm.positionIsFrame: False
```

(Motif and CDE) To add the two entries, you also can edit the following file manually:

```
$HOME/.Xdefaults
```

Notice to Sun SPARCstation 5 Users

A bug exists in some early revisions of the SPARCstation 5. This bug can cause LabVIEW and other programs to hang the system when executing certain floating-point operations. When this condition occurs, you must physically reset the computer to recover. The problem exists in the firmware of the computer and can occur when running SunOS 4.1.3_U1, SunOS 4.1.4, and Solaris 2.*x*.

Note ***This bug has been reported only on early revisions of the 70 MHz and 85 MHz SPARCstation 5.***

To determine whether your SPARCstation 5 is affected, perform the following steps.

Caution ***Following these steps temporarily interrupts the operation of your computer, so you should warn anyone who might be using your computer remotely.***

1. From your SPARCstation 5 console, hold down the `<Stop/L1>` key (located near the upper left corner of your keyboard) and press the `<A>` key to break into the PROM monitor.
2. One of the following two prompts appears:

   `Type b (boot), c (continue), or n (new command mode)>`

   `Type 'go' to resume ok`

   In the first case, select `n` to go to new command mode, where you see an **`ok`** prompt. If you already have an **`ok`** prompt, skip to step 3.

3. At the **ok** prompt, type:

   `module-info`

   You then see information similar to the following lines:

   ```
   CPU FMI,MB86904 Rev. 2.5 : 70.0 MHz
   SBus (Divide By 3)       : 23.3 MHz
   ```

4. Type `go` to exit the monitor and resume operation of your system.

If your CPU Revision number (2.5 in this example) is earlier than 3.2, *and* your CPU clock speed (70.0 MHz in this example) is less than 110 MHz, then your computer has this problem. Contact Sun and ask to have your CPU firmware upgraded to swift_pg 3.2 or later. (Swift is the code name used by Sun for the SPARCstation 5 firmware.) The Sun Bug ID number for this problem is 1151654.

If you have a SPARCstation 5 with this bug, National Instruments strongly recommends upgrading your firmware.

☞ **Note** ***This problem can affect programs other than LabVIEW. Notably, the GNU C compiler also can produce code that hangs your system in versions prior to 2.6.0.***

Compatibility Issues between Versions 4.1 and 5.*x*

This section describes the compatibility issues between different LabVIEW versions.

Compatibility VIs for New Server Functionality

LabVIEW now can act as a server, so you have expanded control over VIs. You can control VIs across a TCP/IP network and, on Windows, the ActiveX interface. LabVIEW includes Compatibility VIs for the VI Control VIs that exist in previous versions. For information about how to implement the functionality from the VI Control VIs using the new server functions, open each VI Control VI and analyze the implementation of the VI Server feature. You can copy this code to your new LabVIEW applications.

Compatibility VIs for ActiveX Functions

The ActiveX functionality has expanded. The functions are more generic because LabVIEW now can act as an ActiveX server as well as a client. Compatibility VIs are provided for the ActiveX functions that exist in

previous versions. For more information about the new ActiveX functionality, refer to the *Improvements to ActiveX Automation* section in Chapter 2, *New Features in LabVIEW 5.1*.

Additional Resources

The LabVIEW documentation set, including the *Code Interface Reference Manual* and the *VXI VI Reference Manual*, is available in Portable Document Format (PDF) on the LabVIEW CD in the `manuals` directory. You can copy this directory or selected PDF files to the `LabVIEW\manuals` directory on your hard drive. You must have Adobe Acrobat Reader 3.0 or later installed to view these files.

(Windows and Macintosh) If you need to perform data acquisition, read the *LabVIEW Data Acquisition Basics Manual,* which contains important information about using the DAQ VIs and examples you can find in LabVIEW. For reference information about a particular DAQ VI, refer to the *LabVIEW Function and VI Reference Manual* and the *LabVIEW Online Reference*, which you can access by selecting **Help»Online Reference…**. Chapter 2, *New Features in LabVIEW 5.1*, also contains information about new features and VIs.

The DAQ examples folder contains a VI library named `RUN_ME.LLB` that has a Getting Started example VI for analog input, analog output, digital I/O, and counters. The `RUN_ME.LLB` examples give you an excellent starting place for data acquisition.

Upgrading to LabVIEW 5.1

If you are upgrading from a version prior to 5.0, you can find upgrade information on your LabVIEW CD and also on our web site. The *LabVIEW 5.0 Upgrade Notes* are available as an Adobe Acrobat file called `Upgrade.pdf` in the `LabVIEW\manuals` directory. To find this information on our web site `www.natinst.com/support/`, search the Product Manuals Library for the LabVIEW listings, where you will find the *LabVIEW 5.0 Upgrade Notes*.

For more information about features, refer to the *LabVIEW User Manual* and the *G Programming Reference Manual*. LabVIEW also offers extensive online documentation, which you can access by choosing **Help»Online Reference...**.

Converting VIs

Upgrading LabVIEW is an automated process. When you open a VI created in a previous version, LabVIEW automatically converts and compiles the VI.

Conversion is a memory-intensive operation. When LabVIEW loads a VI saved in an earlier version, it loads all components of the converted VI (front panel, block diagram, and data) into memory, then compiles the VI in memory. In addition, LabVIEW loads into memory the components of all subVIs needing conversion.

You can estimate the amount of memory required to convert VIs by totalling the amount of memory your VIs and all of their subVIs occupy on disk. If these VIs are in VI libraries, add approximately 30 percent of the VI library size because the VIs are compressed. The conversion process might require at least that much memory and an additional 3 MB of memory to run LabVIEW.

If your computer does not have enough memory to convert your VIs all at once, convert the VIs in stages, by components. Examine your hierarchy of VIs and begin by loading and saving subVIs in the lower levels of the hierarchy. You then can progress gradually to the higher levels of the hierarchy. You also can choose **File»Mass Compile** to convert a directory of VIs. Notice, however, that this option converts VIs in a directory or VI library in alphabetical order. If a high-level VI is encountered first, **Mass Compile** requires approximately the same amount of memory as if you opened the high-level VI first.

You can monitor your memory usage with the **Help»About LabVIEW...** option, which summarizes the amount of memory you have used.

(Macintosh) Before converting VIs, increase the memory allocated to LabVIEW from the Finder by selecting the LabVIEW icon, then choosing **Windows»Show VI Info...** from the menu.

Upgrading Application Libraries and Toolkits

Most existing toolkits function with LabVIEW 5.1 without problems. However, you need to move the VIs so they appear in the menus. LabVIEW 5.1 is compatible with toolkits designed for 3.0, with the following exceptions.

You must upgrade the following add-ons for compatibility with LabVIEW 5.1:

- **LabVIEW Application Builder**—You must upgrade to LabVIEW Application Builder 5.1. This upgrade is free to existing users of the LabVIEW Application Builder. If you have the Professional Development System, the new version of the application builder libraries is included in the installation.
- **Professional G Developers Toolkit**—If you have the Professional G Developers Toolkit, you must upgrade to version 5.1. This upgrade is free to existing users of the Professional G Developers Toolkit. If you have the Professional Development System, the new version of the Professional G Developers Toolkit is included in the installation.
- **LabVIEW Test Executive**—If you use LabVIEW Test Executive 5.0 or earlier, you must upgrade to LabVIEW Test Executive 5.1. This upgrade is free to existing users of LabVIEW Test Executive 5.0.

With minor exceptions, you can use the previous version of the following toolkit with LabVIEW 5.0:

- **Internet Developers Toolkit for G**—You can use the Internet Developers Toolkit 4.1 with LabVIEW 5.0, but you must delete `printvi.llb`, located in the `user.lib\internet\image` directory. Alternatively, upgrade to version 5.0 of this toolkit, which includes this fix and is free to existing users.

The following toolkits do not install VIs in a location that causes them to appear in the palettes. These toolkits are being updated to version 5.0. You can use the existing toolkits by moving VIs to `vi.lib\addons` or `user.lib`. Alternatively, you can choose **Edit»Edit Control and Function Palettes** and add them to the palette of your choice.

- Statistical Process Control Toolkit 1.0
- Proportional-Integral-Derivative (PID) Control Toolkit 1.0—upgrading to version 5.0 of this toolkit is recommended

If you are using the LabVIEW Professional Development System or the Full Development System, you already have two other toolkits—G Math and Picture Control. LabVIEW Base Package users can get these toolkits by upgrading to either of the higher-level LabVIEW development systems.

2

New Features in LabVIEW 5.1

This chapter describes the features added and the significant changes made between versions 5.0 and 5.1. To help you learn more about LabVIEW, version 5.1 offers extensive online documentation, which you can access by choosing **Help**»**Online Reference...**.

Dialog Box, Menu, and Window Enhancements

This section describes changes to LabVIEW dialog boxes, menu items, and behavior or appearance of front panel objects and windows.

Scaling Front Panel Objects

With LabVIEW 5.1, you can designate one particular front panel object or all objects on an entire front panel to scale automatically when the front panel window resizes. If you set a front panel object to scale with the window, the object resizes itself automatically in proportion to any changes in the front panel window size. Also, the other objects reposition themselves to remain consistent with their previous placement on the front panel.

☞ **Note** ***You cannot designate more than one particular object on a front panel to resize automatically. You can designate either one particular object on the front panel, or all objects on the front panel.***

You can set any front panel object to scale automatically when the front panel window is resized. The following list describes important information you should know about scaling objects on the front panel:

- LabVIEW scales objects automatically in the same way you resize the object manually. For example, because you can resize numeric boxes horizontally only, they can scale horizontally only—never vertically.
- When a front panel object is resized, the font size never changes. Thus when an object scales automatically, the font size remains the same.
- You cannot set multiple objects to scale on the front panel unless you set all of them to scale. You can either set *one* object on the front panel to scale automatically, or set *all* objects to scale automatically.

- Once an object scales itself automatically, it might not scale back to its exact original size, when you size the window back to its original position. However, you can use the Undo command to restore the original size.
- When scaling arrays, you can set scaling either on the array itself, or on the objects within the array.
  - When you set scaling on the array, you adjust the number of rows and columns you can see within the array.
  - When you set scaling on the objects within the array, you always see the same number of rows and columns—though different sizes—within the array.

Immediately after you designate an object to scale automatically, several regions on the front panel appear outlined by dotted lines. When you resize a window, objects selected to scale automatically reposition themselves in a manner that is consistent with their previous placement within these regions. To see these regions—to show the dotted lines on the front panel—you must meet the following two conditions:

- You have selected one particular object on the front panel to scale automatically.
- You are operating in **Edit** mode.

Setting an Object to Scale

You can designate any front panel object to scale automatically to match changes in the front panel window.

1. Select the front panel object you want to scale.
2. Choose **Edit»Scale Object With Window**. This option appears with a checkmark beside it when it has been selected.

☞ **Note** ***If you want to set automatic scaling for all objects on the front panel through the* VI Setup *dialog, and you have already set a single object on the front panel to scale, you must first deselect automatic scaling for the particular object. To do this, select the front panel object and then choose* Edit»Scale Object With Window. *The checkmark disappears when you deselect this option.***

Setting All Objects on a Front Panel to Scale

You can designate an entire front panel so that all of its objects scale automatically to match changes to the front panel window.

1. From the front panel of the VI, select **VI Setup**.
2. Under **Window Options**, select **Scale All Objects on Panel**. With this option selected, resizing the front panel automatically causes all objects on the front panel to resize and reposition themselves accordingly.

☞ **Note** ***After you set* Scale All Objects on Panel*, you cannot set or unset scaling on an individual object on the front panel. A dialog box appears that prompts you to choose between setting automatic scaling for either one particular object or all objects on the front panel.***

Defining a Minimum Window Size

You can specify a minimum window size for front panels by following a simple procedure.

1. From the front panel of the VI, select **VI Setup**.
2. Under **Window Options**, go to the **Minimum Window Size** options.
3. Enter the minimum **Width** and **Height**, in pixels, that you want to define for the window.

Maintaining Window Proportions with Monitor Resolution

LabVIEW can maintain its front panel window proportions relative to the resolution of your monitor. When you choose this setting for a VI, the percentage of the screen used by that VI's front panel window stays the same no matter what the end user's screen resolution. Complete the following instructions to activate this option.

1. From the front panel of the VI, select **VI Setup**.
2. Under **Window Options**, select **Maintain Window Proportions with Monitor Resolution**.

Notice that choosing this option only maintains window proportions. To have the objects maintain proportions, you must also choose **Scale All Objects on Panel**. Alternatively, you could choose a single object to scale when the window changes size.

Saving for a Previous Version

With LabVIEW 5.1, you can save your VIs for the previous version of LabVIEW (LabVIEW 5.0). This makes upgrading LabVIEW very convenient, and helps you to maintain your VIs in multiple versions of LabVIEW when necessary. You can upgrade to new versions and always have the capability to go back to your previous version, should you ever need to.

When you choose to save a VI for the previous version, LabVIEW attempts to convert not just that VI, but all the VIs in its hierarchy, excluding `vi.lib` files. Complete the following steps to save a hierarchy of VIs for a previous version of LabVIEW.

1. For the top VI in your hierarchy of VIs, choose **File»Save with Options**. The **Save with Options** dialog box appears.
2. Select the **Save for LabVIEW 5.0.x** option to save your VIs for the previous version.
3. Click **Save**. Immediately after you save, the **Choose a Directory** dialog box appears.
4. Choose the directory where you want to save the VIs.
5. Click **Save**.

Often a VI uses functionality that is not available in the previous version. In such cases, LabVIEW saves as much as it can and produces a report of what could not be converted. The report appears immediately in the **Save for LabVIEW 5.0.x Warnings** dialog box. Click **OK** to acknowledge these warnings and close the dialog box. Click **Save...** to save them to a text file that you can review later.

Accessing Recently Opened Files

LabVIEW 5.1 gives you easy access to recently opened files. To find a file that was open previously, instead of searching through various directories, you can use the **Recently Opened Files** list. This list includes the following file types:

- VI (`*.vi`)
- control (`*.ctl`)
- VI template (`*.vit`)
- control template (`*.ctt`)

When you choose **File»Recently Opened Files**, a submenu appears that contains the list of recently opened files. Select the file name to open it. The files are listed in chronological order, with the most recently opened file listed first. If the list is empty, the **Recently Opened Files** option is dimmed. The **Recently Opened Files** list displays up to 10 file names.

☞ **Note** ***When two or more files have the same file name but reside in different directories, the* Recently Opened Files *list displays the full path.***

Searching in LabVIEW

LabVIEW 5.1 includes new options designed to help you find VIs, subVIs, and text references more easily.

For more information on the **Find** dialog box in LabVIEW, refer to the *Find Dialog Box* topic in the LabVIEW *Online Reference*, or refer to the *Finding VIs, Objects, and Text* section in Chapter 3, *Using SubVIs*, of the *G Programming Reference Manual*.

Find Dialog Box

The **Find** dialog box now includes the **Include SubVIs** option in the **Search Scope** section. The **Include SubVIs** option lets you search for an object or text within the subVIs of the VI you currently have open. By default, this option is not enabled.

Also, the **Search in Hierarchy Window** and **Search VIs in vi.lib** options are now labeled **Include Hierarchy Window** and **Include VIs in vi.lib**, respectively, in LabVIEW 5.1. The functionality of these two options has not changed.

To bring up the **Find** dialog box, select **Project»Find...**, or press <Ctrl-f> **(Windows)**; <command-f> **(Macintosh)**; <meta-f> **(Sun)**; or <Alt-f> **(HP-UX** and **Linux)**.

Find All Instances and Search Results Dialog Box

To make it easier to search for all instances of a VI, LabVIEW now includes the **Find All Instances** pop-up option. You can right-click a subVI or type definition (constant, control, or indicator) to find all instances of that object. Also, you can right-click a connector pane or hierarchy window object to find all instances of that object.

To find all instances of an object, right-click the object for which you want to find references:

- If you are searching for a reference to a type definition, select **Find»All Instances**. (For more information about type definitions, refer to the *Type Definitions* section in Chapter 24, *Custom Controls and Type Definitions*, of the *G Programming Reference Manual*.)
- If you are searching for a reference to a subVI, right-click the VI icon in the block diagram, hierarchy window, or connector pane, and select **Find All Instances**.

If LabVIEW finds one or more references to the object, the **Search Results** dialog box appears (otherwise the object is highlighted).

For more information on the **Find** dialog box in LabVIEW, refer to the *Find Dialog Box* topic in the LabVIEW *Online Reference*, or refer to the *Finding VIs, Objects, and Text* section in Chapter 3, *Using SubVIs*, of the *G Programming Reference Manual*.

Macintosh Navigation Services in LabVIEW

LabVIEW 5.1 takes advantage of the Macintosh Navigation Services and the Macintosh Appearance Manager to give you a more consistent Macintosh user interface under MacOS 8.5.

Navigation Services comes standard with MacOS 8.5. With Navigation Services in LabVIEW 5.1, you can save your VIs and create new directories more easily.

Use Navigation Services with LabVIEW the same way you would with any other Macintosh application. However, LabVIEW 5.1 includes the following customizations to the Navigation Services dialog boxes:

- The **Save** dialog box includes the **Use LLBs** button, which lets you switch to LabVIEW's **File Dialog** dialog box for saving into libraries (`.llb` files).
- The **Append File** dialog box always includes the **New...** button, which lets you create a new folder or file. (Standard Navigation Services behavior displays a **New Folder** button in cases in which you can create new folders only.)

By taking advantage of the Macintosh Appearance Manager under MacOS 8.5, LabVIEW's dialog controls take on the native look and feel of the Macintosh.

New VIs, Functions, and Controls

This section describes the new functionality in LabVIEW 5.1 for advanced mathematics, picture control, report generation, and image management.

Changes to Controls and Indicators

LabVIEW 5.1 has improved the appearance of controls and indicators. Some controls in LabVIEW 5.1 look slightly different than in previous versions when you drop them on the front panel. However, controls on existing VIs you created in previous versions of LabVIEW are not updated. You will notice improvements including the addition of color to many controls, but each of these controls works the same way. The following two sections—*Labels* and *Dialog Controls*—describe the changes to control and indicator labels and the new dialog controls in more detail.

Labels

By default, when you create a new control or indicator, its name label includes the name of the type of control or indicator (such as *Slide* or *String*) and, if necessary, a number distinguishing it from other controls and/or indicators of the same kind. For example, if you place a slide control on the front panel, it is labeled *Slide*. If you place another slide control on the front panel, it is labeled *Slide 2*.

When you replace a control or indicator that has a default label (as described above) with an object of a different type, the control or indicator is renamed as well. For example, if you replace a slide control labeled *Slide 2* with a knob control, the knob appears with the label *Knob*. If a slide or control already exists on the front panel with the label *Knob*, the new knob takes the label *Knob 2* instead.

By default, when you drop an object on the front panel, its name label is highlighted to let you immediately type in a replacement for the default name. Unlike in previous versions of LabVIEW, if you want to hide a name label right after you place it on the front panel, right-click the control or indicator and deselect **Label** from the **Show** menu.

By default, name labels appear transparent. To make a name label appear in a raised box, select **Edit»Preferences**, select **Front Panel** from the drop-down menu, and deselect **Use transparent name labels**.

Dialog Controls

All the dialog controls are now grouped together in a new **Dialog** palette, shown in Figure 2-1 below. You can reach this palette from the front panel by selecting **Controls»Dialog**.

Figure 2-1. Dialog Palette

This palette includes the dialog controls listed below. You can still find these controls in the same palettes as in previous versions of LabVIEW.

- Dialog Numeric Control
- Dialog String Control
- Dialog Ring
- Dialog Button
- Cancel Button
- Dialog Checkbox
- Dialog Radio Button

Also, this palette includes two new controls: the *Dialog Listbox* control and the *Dialog Recessed Frame* control. The Dialog Listbox control behaves like other listboxes, but matches the tabbed highlighting and bordering functionality and appearance of the native system controls on your

platform. The Dialog Recessed Frame control behaves like other decoration objects found in the **Decorations** palette. You can use it as a grouping border to put around other dialog controls.

3D Graph Controls for Windows

LabVIEW 5.1 adds a new way to represent data on your front panel: the 3D graph. The LabVIEW 3D graph uses ActiveX technology and new VIs that handle three-dimensional representation. You can set parameters for the 3D graphs VIs to change behavior at runtime, including setting basic, axes, grid, and projection properties.

For more detailed information about the 3D graphs VIs see the *3D Graphs VIs* topic in the LabVIEW *Online Reference*.

You can find examples of the 3D graphs VIs in the `Examples\General\graphs\3dgraph.llb` directory.

☞ **Note** ***The 3D Graph controls are available for Windows only in the LabVIEW Full Development System and Professional Development System.***

Enhancements to Property and Invoke Nodes

With LabVIEW 5.1, VI Server and Application Class property and invoke nodes do not always need a wired refnum input or output. This makes it easier to program many common VI Server functions because you can drop fewer diagram objects.

If you do not wire inputs or outputs, LabVIEW uses default values. The default input value for the Application Class is your local LabVIEW. The default value for the Virtual Instrument Class is your current VI—the VI in which you have placed the property or invoke node.

Improvements to VI Server Properties in Reserved VIs and Runtime Systems

In LabVIEW 5.1, the VI Server has been enhanced so you can set many more properties while a VI is running or in run-time versions. These properties are listed below. Also, the LabVIEW Help Window now includes thorough explanations for each Virtual Instrument Class property and any limitations they might have. Choose **Help»Show Help** to open the Help Window. You can move your cursor over the property portion of any Virtual Instrument Class property node, and documentation for that property appears in the Help Window.

With LabVIEW 5.1, in addition to the VIs that previously worked at runtime, you can set any of the following properties on running VIs and in run-time versions:

- AutoLogging»Log File Path
- AutoLogging»Log at Finish
- AutoLogging»Print at Finish
- Edit Mode on Open
- Execution»Close after Call
- Execution»Show Front Panel on Call
- Front Panel Window»Allow Runtime Popup
- Front Panel Window»AutoCenter
- Front Panel Window»Closeable
- Front Panel Window»Highlight Return Button
- Front Panel Window»Is Dialog
- Front Panel Window»Resizeable
- Front Panel Window»Show Menu Bar
- Front Panel Window»Show Scroll Bars
- Front Panel Window»Size to Screen
- Front Panel Window»Title Bar Visible
- Help»Document Path
- Help»Document Tag
- Tool Bar»Show Abort Button
- Tool Bar»Show Free Run Button
- Tool Bar»Show Run Button
- Tool Bar»Visible
- VI Description

You also can set any of the following properties on running VIs in LabVIEW 5.1 (but not in run-time versions):

- Execution»Show Front Panel on Load
- Execution»Run When Opened
- Execution»Suspend on Call
- History»Always Add Comments at Save
- History»Prompt for Comments at Close

- History»Prompt for Comments at Save
- History»Record Application Comments
- History»Use Defaults

Ring Enhancements

All ring and enum constants include a down arrow, which distinguishes them from numeric constants. Also, rings and enums on panels or block diagrams have scroll bars when the menus include many items. LabVIEW 5.1 menus, like listboxes, now support type completion. Just type the first few characters and LabVIEW finds a matching item in the list. Use the <Tab> and <Shift-Tab> keys to move to the next and previous matching item, respectively.

Figure 2-2 shows an example of a ring constant with many items in its menu.

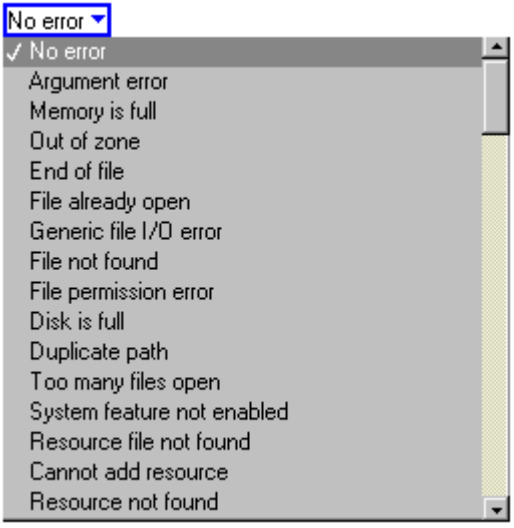


Figure 2-2. Ring Constant Examples

IVI Instrument Drivers and ActiveX

Some parameters for ActiveX and IVI take a discrete list of valid values. Previously, when building ActiveX and IVI applications, you used numeric values to set parameters for properties and methods that control devices or programs. You had to enter a particular numeric value in a numeric control or constant, which required you to know which numeric values corresponded to which settings.

With LabVIEW 5.1, you are not required to know these values. You can choose from a descriptive name in a ring to set parameter values. The selections available in the ring depend on the refnum passed to the node. See Figure 2-3 and Figure 2-4 below for examples of using these numeric and ring constants.

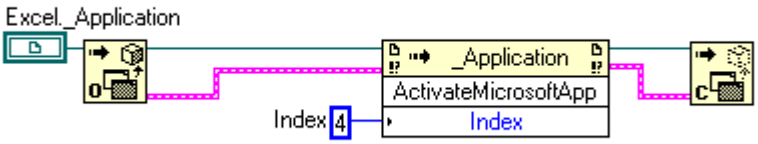


Figure 2-3. Setting a Data Value with a Numeric Constant

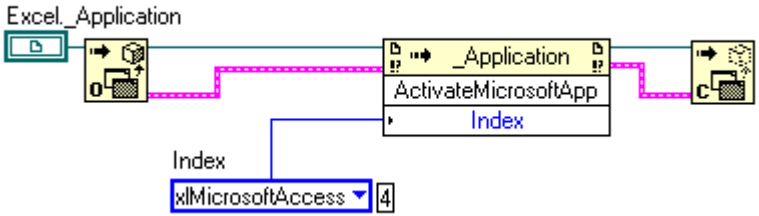


Figure 2-4. Setting a Data Value with a Ring Constant

In both examples, the Microsoft Excel application is accessed and a method is invoked. The **Index** parameter has several options: *MicrosoftAccess*, *MicrosoftFoxPro*, *MicrosoftMail*, *MicrosoftPowerPoint*, *MicrosoftProject*, *MicrosoftSchedulePlus*, and *MicrosoftWord*.

Previously you were required to know that the **Index** parameter corresponding to the *MicrosoftAccess* option is the data value 4. Now you can select the *Microsoft Access* option from the drop-down menu in the ring constant.

To access the ring constant when building an ActiveX or IVI application, right-click the parameter that accepts data values and select **Create Constant**. To see the corresponding numeric data value, right-click the ring constant and select **Show»Digital Display**.

ActiveX Enhancements for Windows

This section describes the ActiveX enhancements in LabVIEW 5.1.

Ring Enhancements

With LabVIEW 5.1, you do not have to use numeric values when setting parameters for properties and methods that control devices or programs. You can use a ring constant as well. See the previous section, *IVI Instrument Drivers and ActiveX*, for more information.

Support for ActiveX Events

LabVIEW 5.1 lets you access events associated with ActiveX objects.

Now, in addition to accessing the properties and methods associated with an ActiveX object when building an ActiveX server VI, you also can access the events. You can embed a control on a VI front panel, then execute code based on an event that occurs. For example, you can place a tree view control that list the contents of a directory. Using events, you can specify that when the user double-clicks on a particular file, the contents of that file open.

You build VIs that accept events using the ActiveX Event functions, which you find in the **Functions»Communications»ActiveX»ActiveX Events** palette.

Working with ActiveX Events

This is the basic procedure for creating a VI designed to create and wait on an ActiveX event queue, then destroy the event queue. An *event queue* is a tag that corresponds to an internal list of events a control receives.

Note ***If you generated an Automation Refnum using an Automation Client function, omit Step 1.***

1. On the front panel, select the Container control found on the **Control»ActiveX** palette.
2. Create an ActiveX event queue using the Create ActiveX Event Queue VI (described below).

3. Wire the Automation refnum from the container terminal or an Automation function to the Create ActiveX Event Queue VI.
4. Place the Wait On ActiveX Event VI on the block diagram.
5. Wire the event queue to the Wait On ActiveX Event VI.
6. Dispose of the event queue using the Destroy ActiveX Event Queue VI.

ActiveX Event Functions for Windows

You can use the other ActiveX Server Event functions to pass an event queue from one task to another separate task. Figure 2-5 shows the ActiveX event functions palette, which you access from the block diagram through **Functions»Communications»ActiveX»ActiveX Events**.

Figure 2-5. ActiveX Events Palette

For more detailed information about the ActiveX event functions see the *ActiveX Event Functions* topic in the LabVIEW *Online Reference*.

Improvements to ActiveX Automation

The Automation Open function now includes the optional **machine name** string input. Use **machine name** to specify on which computer you want to open an Automation Server object. If you do not specify a machine name, LabVIEW creates the object on the local machine. See the Open Automation Refnum description in Chapter 51, *ActiveX Automation Functions*, in the *LabVIEW Function and VI Reference Manual*.

Also, LabVIEW 5.1 supports enumerations in the ActiveX Invoke and Property Nodes. When you right-click an Invoke Node or Property Node's terminal that is part of an enumeration or is a constant, you create a ring with the available enumeration values. See *Ring Enhancements*, earlier in this chapter, for more information.

HiQ and MATLAB Functionality for Windows

HiQ and MATLAB are software packages that help you organize and visualize real-world math, science, and engineering problems. You can use HiQ and MATLAB to express numeric formulas elegantly. Now with LabVIEW you can load and edit HiQ and MATLAB scripts into your block diagram code so LabVIEW works with their advanced mathematics functionality.

☞ **Note** ***You must have HiQ 4.1 or MATLAB 5.0 or later installed to use this new feature successfully. You can install HiQ 4.1 from the LabVIEW CD-ROM. HiQ and MATLAB functionality is available for Windows only in the LabVIEW Full Development System and Professional Development System.***

There is a new script node that works similarly to the formula node. You can find a HiQ script node and a MATLAB script node in the **Functions»Mathematics»Formula** palette. You can place a script node on your block diagram and enter a script according to the syntax of HiQ or MATLAB. LabVIEW then communicates with that script server engine for you. When you create inputs and outputs on the script node, those values are passed between HiQ or MATLAB and LabVIEW. If you already have a script written, you can import it from HiQ or MATLAB. See the section *Importing or Exporting a Script* for more information.

Although the basic functionality of the script node is similar to that of the formula node, the script node can handle more datatypes than the formula node. This script node supports the same HiQ or MATLAB datatypes as supported by ActiveX.

☞ **Note** ***Because of the nature of the HiQ and MATLAB script languages, you must choose which LabVIEW datatype each terminal should be. For more information, see the section Configuring the Datatype of a Terminal, later in this chapter.***

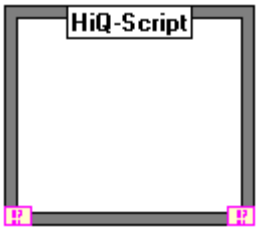
The following two sections, *Creating a HiQ Script* and *Creating a MATLAB Script*, contain procedures for creating a script that does the following:

1. Generates a matrix of random values
2. Plots that information to a graph
3. Displays the graph in either product
4. Presents the generated random values on your front panel in LabVIEW

Creating a HiQ Script

Complete the following instructions to create and run a VI that uses a HiQ script.

1. From the block diagram, choose **Functions»Mathematics»Formula»HiQ-Script**. Place the node on the block diagram, and size it according to the amount of script you want to include in the window. The HiQ script node appears as shown below.



2. With the operating tool, enter the script in the HiQ script node. For example, the following simple HiQ script creates a matrix of random values, plots that information to a graph, and displays the matrix in HiQ.

```
a = random({50, 50});
g = createGraph(a);
createView(g, true);
```

You can use a HiQ script window to edit, compile, and run your script directly from HiQ to ensure that your script behaves properly.

3. Right-click the HiQ script node and select **Edit In Server**. This action launches HiQ, and a script window appears containing your HiQ script.
4. Within HiQ, right-click and choose **compile** from the pop-up menu. A message window appears telling you of any compile time errors. You do not have to compile the script explicitly; HiQ compiles the script automatically when you run it.
5. Right-click and choose **Run** from the pop-up menu. A message window informs you of any runtime errors.

To access HiQ context-sensitive online help, place the cursor inside any function and press <F1>. You can use this information to help you build your script.

6. Close the HiQ window to update and return to the HiQ script node on the LabVIEW block diagram.

7. To add inputs and outputs for variables, right-click the right side of the node frame and select **Add Input** or **Add Output**. Type in `a` to add an output for the **a** variable that is included in your HiQ script. By default, your node already includes one input and one output terminal for the **error in** and **error out** parameters.
8. Verify the datatype of the inputs and/or outputs. The error-checking input and output are already the correct datatype. In HiQ, the default datatype for any new input or output is Real. Right-click the **a** output and select **Choose Type**. From the submenu that appears, select an available datatype: Integer, Real, Complex, Text, Integer Vector, Real Vector, Complex Vector, Integer Matrix, Real Matrix, and Complex Matrix. For the **a** output, choose **Real Matrix**.
9. Create controls and indicators for each input and output. Right-click the **a** output terminal and select **Create Indicator**. Right-click the **error out** output terminal and select **Create Indicator**. Indicators for **a** and **error out** appear on the front panel, and terminals appear wired to these outputs on the block diagram.
10. Go to the front panel. Resize your **a** indicator so you can see the generated numbers when you run the VI.
11. Run the VI. LabVIEW launches HiQ and a new HiQ window appears, labeled *G in Notebook1*, that displays the matrix. The values that make up this matrix are displayed in the **a** indicator of your front panel in LabVIEW.

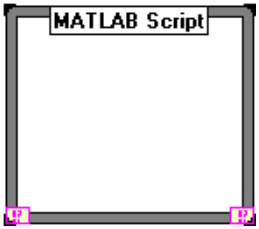
The second parameter in the CreateView function specifies whether to pause execution of the HiQ script while the view is visible. The HiQ script in Step 2 does not finish until you click the **Continue** button in the HiQ window to dismiss the view.

12. Change the `true` parameter to `false` and re-run the VI to see the difference.

Creating a MATLAB Script

Complete the following instructions to create and run a VI that uses a MATLAB script.

1. From the block diagram, choose **Functions»Mathematics»Formula»MATLAB Script**. Place the node on the block diagram, and size it according to the amount of script you want to include in the window. The MATLAB script node appears as shown below.



2. With the operating tool, enter the script in the MATLAB script node. For example, the following simple MATLAB script creates a matrix of random values, plots that information to a graph, and displays the matrix in MATLAB.

```
a=rand(50)

surf(a)
```

3. To add inputs and outputs for variables, right-click the right side of the node frame and select **Add Input** or **Add Output**. Type in `a` to add an output for the **a** variable that is included in your MATLAB script. By default your node already includes one input and one output terminal for the **error in** and **error out** parameters.
4. Verify the datatype of the inputs and/or outputs. The error-checking input and output are already the correct datatype. In MATLAB, the default datatype for any new input or output is Real. Right-click the **a** output and select **Choose Type**. From the submenu that appears, select an available datatype: Real, Complex, Real Vector, Complex Vector, Real Matrix, or Complex Matrix. For the **a** output, choose **Real Matrix**.
5. Create controls and indicators for each input and output. Right-click the **a** output terminal and select **Create Indicator**. Right-click the **error out** output terminal and select **Create Indicator**. Indicators for **a** and **error out** appear on the front panel and terminals appear wired to these outputs on the block diagram.
6. Go to the front panel. Resize your **a** indicator so you can see the generated numbers when you run the VI.

7. Run the VI. LabVIEW launches MATLAB and a new MATLAB window appears, labeled *Figure No. 1*, that displays the matrix. The values that make up this matrix are displayed in the **a** indicator of your front panel in LabVIEW.

Importing or Exporting a Script

Complete the following steps to import a script into a script node in LabVIEW.

1. Right-click the HiQ or MATLAB script node.
2. Select **Import…**.
3. Choose the file you want to import and click **Open**. The script text appears in your node.

Complete the following steps to export a script to LabVIEW from MATLAB or HiQ.

1. Right-click the HiQ or MATLAB script node.
2. Select **Export…**. The **Save Script as** dialog box appears.
3. Enter the name you want to save your file as or choose the file you want to overwrite.
4. Click **Save**.

☞ **Note** ***HiQ and MATLAB script files are actually text files. Text files usually have the extension*** `.txt`***. However, HiQ files have a*** `.hqs` ***extension and MATLAB files have a*** `.m` ***extension.***

Choosing a Script Server

Complete the following steps to change the server behind your script node.

1. Right-click the HiQ or MATLAB script node.
2. Select **Choose Script Server»HiQ-Script** or **Choose Script Server»MATLAB Script** depending on whether you want HiQ or MATLAB.

Some datatypes in HiQ are not recognized by MATLAB and vice versa. If you change the script server from one of these mathematics products to the other after you have assigned datatypes, these datatypes do not change automatically. The VI will be broken. When you click the broken arrow in the toolbar of your VI, the **Error List** window reports these incorrect datatypes.

Scrolling through a HiQ or MATLAB Script

Just as with a regular formula node, you can display a scrollbar within your script node by right-clicking the node and selecting **Show»Scrollbar**. A scrollbar appears on the right side of your script node.

Configuring the Datatype of a Terminal

Both HiQ and MATLAB are loosely typed script languages and do not determine the datatype of a variable until after the script executes. Therefore LabVIEW cannot determine a variable's type in **Edit** mode. However, LabVIEW does query the script server to find out possible datatypes, and lets you choose which LabVIEW datatype each terminal should be.

☞ **Note** ***If you do not correctly configure a variable's datatype, LabVIEW will produce either an error or incorrect information at runtime.***

Complete the following steps to change the datatype of an input or output terminal on a script node.

1. Right-click the terminal of the input or output. A pop-up menu appears.
2. Select **Choose Type**. A list of the available datatypes appears, depending on whether you use HiQ or MATLAB.
3. Choose the preferred datatype.

☞ **Note** ***LabVIEW recognizes all the datatypes that HiQ and MATLAB can use, although they might be named differently.***

Table 2-1 shows LabVIEW datatypes and their corresponding datatypes in HiQ and MATLAB.

Table 2-1. HiQ and MATLAB Datatypes in LabVIEW

LabVIEW Datatype	HiQ Datatype	MATLAB Datatype
	Integer	N/A
	Real	Real
	Text	N/A
	Integer Vector	N/A
	Real Vector	Real Vector
	Integer Matrix	N/A
	Real Matrix	Real Matrix
	Complex	Complex
	Complex Vector	Complex Vector
	Complex Matrix	Complex Matrix

Debugging a HiQ or MATLAB Script

Use the following programming techniques to make debugging your script easier:

- Write your script and run it within the native engine (HiQ or MATLAB) for testing and debugging purposes before you import it into LabVIEW. In HiQ, right-click the script node and select **Edit In Server** to debug in a native HiQ script window.
- Verify your datatypes. When you create a new input or output, make sure that the datatype of the terminal is correct. Also, create controls and indicators for your inputs and outputs so you can monitor what values are being passed between LabVIEW and the native engine. For more information about this topic, see the *Configuring the Datatype of a Terminal* section.
- Take advantage of the error-checking parameters for debugging information. Create an indicator for the **error out** terminal on a script node before you run any VI so you can view the generated error information at runtime.

Error Codes

The following table explains error messages you might receive at runtime when working with a script node.

Table 2-2. Script Node Runtime Error Messages

Error Code	Error Code Message	Description
1046	`edScriptCantInitServer`	LabVIEW failed to initiate a new session with the server.
1047	`edScriptCantSetValue`	LabVIEW failed to set the value of a variable to the server.
1048	`edScriptCantGetValue`	LabVIEW failed to retrieve the value of a variable from the server.
1049	`edScriptCantSetScript`	LabVIEW failed to set a script to the server.
1050	`edScriptExecError`	LabVIEW encountered an execution problem. The server returns a string to report the problem.

Integration of Mathematics and Signal Processing VIs

LabVIEW 5.1 includes more VIs designed to help you solve advanced mathematics problems. You can use these VIs to manipulate mathematical formulas on the LabVIEW front panel, and then assemble your problem-solving program graphically on the LabVIEW block diagram.

Previously, much of this functionality was available only with the G Math Toolkit, which has been integrated into the LabVIEW Full Development System.

☞ **Note** ***This functionality is available on all platforms and versions with the exception of the Windows base version.***

Integrating these VIs restructures the LabVIEW **Functions** palette. The new **Mathematics** palettes includes the new Mathematics VIs, as well as some of the VIs and functions formerly found in the **Analysis** and **Structures** palettes.

The **Analysis** palette found in LabVIEW 5.0 has been replaced with the **Signal Processing** palette. For information on the Signal Processing VIs, see the *Signal Processing Palette* section.

For function and reference information about the Mathematics VIs and functions, refer to the *Mathematics VIs* topic in the LabVIEW *Online Reference*.

Mathematics Palette

Table 2-3 lists the subpalettes you find in the **Functions»Mathematics** palette, and lists the previous location of the VIs or functions in LabVIEW 5.0.

Table 2-3. Mathematics VIs Current and Previous Locations

Mathematics Subpalettes	Previously Located in...
Formula	G Math Toolkit and **Structures** palette
1D and 2D Evaluation	G Math Toolkit
Calculus **Differential Equations** subpalette	G Math Toolkit
Probability and Statistics **Probability** subpalette **Analysis of Variance** subpalette	**Analysis** palette
Array Operations	**Analysis** palette
Curve Fitting	**Analysis** palette
Linear Algebra **Advanced Linear Algebra** subpalette **Complex Linear Algebra** subpalette	**Analysis** palette
Zeros	G Math Toolkit, **Analysis** palette
Optimization	G Math Toolkit
Numeric Functions	G Math Toolkit

You can find examples of the Mathematics VIs in the `Examples\Math\math.llb` directory.

Signal Processing Palette

For function and reference information about the Signal Processing VIs, refer to the *Signal Processing VIs* topic in the LabVIEW *Online Reference*.

Table 2-4 lists the subpalettes you find in the **Functions»Signal Processing** palette, and lists the previous location of the VIs or functions in LabVIEW 5.0.

Table 2-4. Signal Processing VIs Current and Previous Locations

Signal Processing Subpalettes	Previously Located in...
Signal Generation	**Analysis** palette
Time Domain	**Analysis** palette (These functions were found in the Analysis palette's **Signal Processing** subpalette)
Frequency Domain	**Analysis** palette and G Math Toolkit (Combines the FFT and Power Spectrum VIs found in the Analysis palette's **Signal Processing** subpalette and the VIs found in the G Math Toolkit's **Transforms** palette.)
Measurement	**Analysis** palette
Filters	**Analysis** palette
Windows	**Analysis** palette

You can find examples of the Signal Processing VIs in the `Examples\Math\sig_proc.llb` directory.

Integration of the Picture Control VIs

With LabVIEW 5.1, you can display complex images and graphs using functionality that previously was included only in the Picture Control Toolkit. Because the Picture Control Toolkit has been integrated into the LabVIEW Full Development System, you can use any of the VIs in the Picture Control VI Library. The Picture Control VIs include drawing operations you can use to create diagrams and build images dynamically. You can create new front panel displays such as specialized bar graphs, pie charts, or Smith charts. You also can display and animate arbitrary objects such as robot arms, test equipment, or a two-dimensional display of a real-world process. For detailed information about these VIs, refer to the *Graphics and Sound VIs* topic in the LabVIEW *Online Reference*.

☞ **Note** ***The Picture Control VIs are available on all platforms and versions with the exception of the Windows base version.***

You can find examples of the Picture Control VIs in the `Examples\Picture` directory.

Sound VIs for Windows and Macintosh

You can use the Sound VIs to integrate sound into your VIs. Some of the things you can do are create a beep to alert the user of an error, play a `.wav` file, and read and write sound data.

For more information, refer to the *Sound VIs* topic in the LabVIEW *Online Reference*. To access the Sound VIs, select the **Functions»Graphics & Sounds»Sound** palette from the block diagram. You can find examples of the Sound VIs in the `Examples\Sound\sndExample.llb` directory.

☞ **Note** ***The Sound VIs are not available on the UNIX operating systems.***

Generating Reports in LabVIEW for Windows

☞ **Note** ***You can use the report generation functions on 32-bit Windows operating systems only (Windows 95/98/NT). This functionality is based on ActiveX technology, which is not available on Macintosh and UNIX platforms. On Macintosh and UNIX platforms, refer to Chapter 5,*** *Printing and Documenting VIs****, in the*** *G Programming Reference Manual****.***

(**UNIX**) Use the System Exec VI to print a file through a command line function. The VI is located in **Functions»Communication**.

(**Macintosh**) You can use the AESend Print Document VI to direct your applications to print a document. The VI is located in **Functions»Communication»Apple Event**.

On Windows platforms, you can create reports of any text-based information a VI generates or the user enters into a string parameter, or you can create a report from an array of 2D numbers. For example, if you develop a LabVIEW program that tests functions, you can create a text-based report of which functions passed and which failed.

To create reports using the Report Generation VIs, go to the **Functions»Report Generation** palette. You can use the Report Generation VIs to do the following:

- Set up a report's headers and footers (which can include date and time stamps)
- Set character font, size, style, and color
- Set a report's margins and tabs
- Determine what information appears on a particular line or page of a report
- Set a report's orientation on a page (lengthwise or widthwise)
- Include text from other files in a report
- Clear information from an existing report (to re-use the report's formatting in a new report)
- Automatically print a report
- Dispose of a report after it is printed (which saves memory)

Hints for Generating Reports in LabVIEW

Make sure any information you want to print is formatted into a string (unless you want to print a 2D array of numbers as a table). If you have a lot of information and you want to write it to a report, you likely will use one or more of the String functions. See Chapter 6, *String Functions*, in the *LabVIEW Function and VI Reference Manual*, for information on the String functions and how they format data.

Use the Easy Text Report VI for less-complicated reports. The following section, *Easy Text Report VI Overview*, contains more information on this VI.

Easy Text Report VI Overview

The Easy Text Report VI creates a basic report from a block of text. This VI is appropriate if you do not need to have fine-grain control over all aspects of the report, and if you do not have a lot of varying data or a numeric table in the report.

With the Easy Text Report VI, you can specify the text font, set up the header and footer, set margins, specify a printer, and set the page's orientation. However, you cannot control where information is placed, append information from another file, or clear the report of font styles, headers and footers, or text. The Easy Text Report VI disposes of the report automatically, which frees up memory space.

See the Easy Text Report VI description in the *Report Generation VI Descriptions* section later in this chapter for information on this VI's parameters.

Tip ***See the block diagram of the Easy Text Report VI for a good example of how you can use the other VIs in the Report Generation palette to create a report.***

Report In/Report Out Parameters

The **Report in** and **Report out** parameters link the VIs used to create a report. All of the Report Generation VIs have both these parameters, with the following exceptions:

- Easy Text Report VI has neither of these parameters.
- Dispose Report VI has only the **Report in** parameter.
- New Report VI has only the **Report out** parameter.

Use these parameters to link a report to the VIs that control a report's appearance, data, and printing. These parameters give you the flexibility to add and control several different features on one report.

Tokens

Tokens are strings you can enter that generate information automatically. You can use tokens to number pages and place a timestamp on the report. These especially are useful in the headers and footers. There are several tokens you can enter in any parameter that takes a string. For example, if you are using the Easy Text Report VI, entering the token `<page>` in one of the footer strings causes a page number to appear in the footer of every page of the report.

Table 2-5 lists some of the more useful tokens.

Table 2-5. Token Descriptions

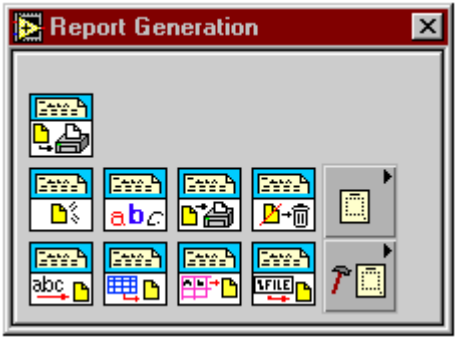
Token	Description
`<page>`	Current page number
`<pages>`	Total number of pages
`<pagenofm>`	Current page number along with the total number of pages in the report. Example: 7 of 30

Table 2-5. Token Descriptions (Continued)

Token	Description
<shortdate>	Current date in the form *xx*/*xx*/*xx*. Example: 10/5/98 The month, date, and year order defaults to the date settings of your operating system.
<longdate>	Current date in the form *Day, Month and Date, Year*. Example: Monday, October 05, 1998 The month, date, and year order defaults to the date settings of your operating system.
<time>	Current time in the form *Hour:Minute:Second AM/PM*. Example: 1:58:22 PM. This token defaults to the clock settings on your computer (that is, whether it uses a 12-hour or 24-hour clock).

Report Generation VI Descriptions

This section describes the Report Generation VIs, including descriptions and data types of each parameter. From the block diagram, you can reach the new Report Generation palette by selecting **Functions»Report Generation**.



The Report Generation palette has two subpalettes:

- **Functions»Report Generation»Report Layout**
- **Functions»Report Generation»Advanced Reports**

You can find examples of the Report Generation VIs in the `Examples\Reports\TextReportExample.llb` directory.

Report Generation Parameter Descriptions

This section describes the parameters used by the Report Generation VIs.

append on new line? (F), if the `True` value is selected, appends the information onto a new line in the report. The default value is `False`.

center footer text is the information you want to appear in the center portion of the footer.

center header text is the information you want to appear in the center portion of the header.

Column Headers determines how each column is labeled in the table.

Column width defines the width of each column in the report's table. The value you enter is in inches or centimeters, depending on the settings you enter in **measurement system** or **measurement system for columns**. The default value is 1.

error in

error out

file path is the path of the text file from which you want to pull information into your report. You must include the file's path in this parameter.

Font Settings allows you to set the font of your report.

Charset sets the character set used in the report (such as the set for a specific language).

Weight sets the characters' weight.

Name specifies the font used in the report. You can type in the name of any font available on your system.

Note ***The name you enter must match the font name exactly.***

Size specifies the size of the font in number of points.

Font Settings indicates what font settings currently are used in a report.

Charset indicates the character set used in the report (such as the set for a specific language).

Weight indicates how dark bold characters appear.

Name indicates the font used in the report.

Size indicates the size of the font in number of points.

Footers allows you to set up what information appears in the footer of each page of the report. You can type in text, leave the parameters blank, or type in a token (see *Tokens*, earlier in this chapter, for information on the tokens you can enter).

left is the information you want to appear in the left side of the footer.

center is the information you want to appear in the center portion of the footer.

right is the information you want to appear in the right side of the footer.

format string determines how the numbers appear in each cell of the table, such as `01`, `1`, `1.0`, or `1.00`.

It is as defined by Format & Append:

```
[Str]%[-][0][Width][.Precision]Conversion[Str]
```

where `-` causes left justification and `0` pads with zeros.

Conversions: d (decimal), x (hexadecimal), o (octal), f (fractional), e (scientific), or g (scientific).

For example, if you wanted the numbers to appear in each as 1.00, you would use the default value: %.2f.

Refer to the *Format & Append and Format Strings Overview* topic in the LabVIEW *Online Reference* for more information on values you can enter in **format string**.

Headers allows you to set up what information appears in the header of each page of the report. You can type in text, leave the parameters blank, or type in a token (see *Tokens*, earlier in this chapter, for information on the tokens you can enter).

left is the information you want to appear in the left side of the header.

center is the information you want to appear in the center portion of the header.

right is the information you want to appear in the right side of the header.

left footer text is the information you want to appear in the left side of the footer.

left header text is the information you want to appear in the left side of the header.

Margins set where the information is placed on each page of the report. You set the margins with the following parameters:

left or **LeftMargin** sets the report's left margin in inches or centimeters.

right or **RightMargin** sets the report's right margin in inches or centimeters.

top or **TopMargin** sets the report's top margin in inches or centimeters.

bottom or **BottomMargin** sets the report's bottom margin in inches or centimeters.

measurement system has three options: **US**, which sets the margins in inches; **Metric**, which sets the margins in centimeters; and **Default**, which sets the margins in the measurement system set up on your computer.

measurement system for columns determines whether the value you enter in **column width** is in inches or centimeters. There are three options: **US**, which sets the columns in inches; **Metric**, which sets the columns in centimeters; and **Default**, which sets the columns in the measurement system set up on your computer.

Number of Copies (1) specifies how many copies of the report you want to print. If you do not specify a number, one copy is printed.

Numerical Data is the numerical information included in the table. The information must be a 2D array.

orientation specifies how the report appears when it is printed. You can choose from **portrait**, which prints the report widthwise, or **landscape**, which prints the report lengthwise.

Printer Name specifies the name of the printer that you want to print this report. If you do not enter a printer name, then this VI uses the default printer set up on your computer. If you do specify a printer name, you must have that printer configured for use with your computer. If you do not specify a printer, you must have a default printer specified on your system.

Report in links a report to the VIs used to control a report's appearance, data, and printing. This holds the report before the VI generates new data for the report.

Report out links a report to the VIs used to control a report's appearance, data, and printing. This holds the report after the VI generates new data for the report.

right footer text is the information you want to appear in the right side of the footer.

right header text is the information you want to appear in the right side of the header.

Row Headers determines how each row is labeled in the table.

Rows/Page determines how many rows appear on each printed page of the report. If set to `0`, as many rows as can fit on the page are printed and column headers are not repeated on each page.

Separate Page? Places the table on a new page of the report.

Text is the information you want to include in the report. Any information you want to include must be in a string.

Text Color (unchanged) lets you select the color of the text in the report. You can use the color box constant with this parameter (which you access from the **Functions»Numeric»Additional Numeric Constants** palette).

Text Color indicates the color currently used for the text in the report.

Text Data is the information you want to be printed in tabular form. If you are using a table control, pass the table's value to this parameter.

Text Options specifies how the text appears in the report.

Italic determines whether subsequent text appears in italics in the report.

Strike Through determines whether subsequent text appears with strikethroughs in the report.

Underline determines whether subsequent text appears underlined in the report.

Bold determines whether subsequent text appears bold in the report.

Text Options indicates what text options currently are set in a report.

Italic indicates whether Italics currently are used in the report.

Strike Through indicates whether strikethroughs currently are used in the report.

Underline indicates whether underlines currently are used in the report.

Bold indicates whether bold currently is used in the report.

Text to be Printed is the information you want to include in the report. Any information you want to include must be in a string.

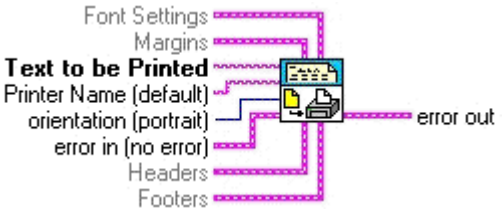
width sets the report's tabs width in inches or centimeters. If you do not enter a tab length, the default value .25 is used.

Easy Text Report

Allows you to pass in a block of text (such as a log) along with optional formatting information, then print the report to a designated printer. (If you do not specify a printer, the default printer is used.)

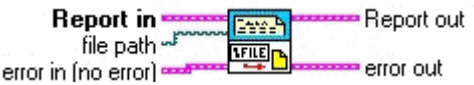
Note ***You cannot use this VI with any other NI Report VIs. Also notice that this VI does not give you fine-grain control over formatting your report. For more complicated reports (such as those with lots of different kinds of information), use the other VIs found in the*** **Report Generation** ***palette.***

You might find using tokens with this VI helpful, particularly in the **Headers** and **Footers** parameters. For example you might want to give the report a time stamp. See the *Tokens* topic, earlier in this chapter, for more information on some common tokens for this VI.



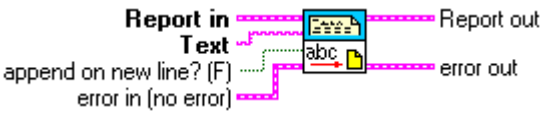
Append File to Report

Appends the text from a text file (`.txt`) into the current report. You must wire the **file path** (including its path) to the VI. The text is appended within the report.



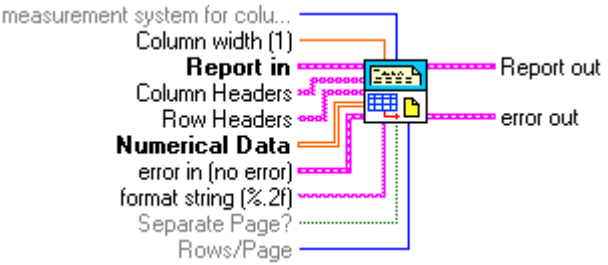
Append Report Text

Appends text to the selected report. The input into **Text** must be a string. The selected report is the one passed into **Report In**. You can append the text to the current position of the cursor in the report or on a new line.



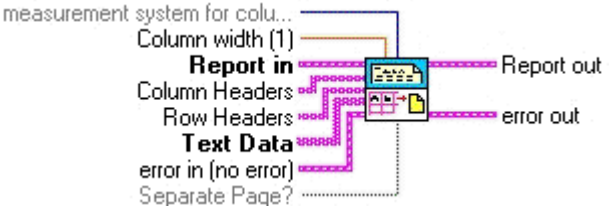
Append Numeric Table to Report

Takes a 2D array of numbers and appends it to a report as a table with the given column widths.



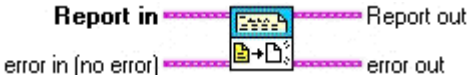
Append Text Table to Report

Takes a 2D array of strings and appends it to a report as a table with the given column widths.



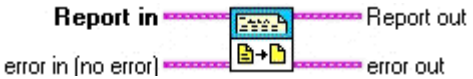
Clear Report

Clears the report of all text, headers, footers, and formatting information.



Clear Report Text

Clears the text and related formatting information from the report.

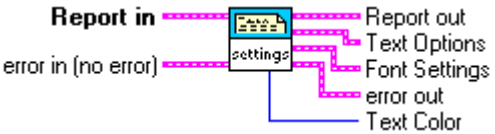


Dispose Report

Closes the report and releases its interface, which saves memory. No further operations are permitted on the report. However, you can create a new report. You can use this VI only as the last VI in the report function you are creating.

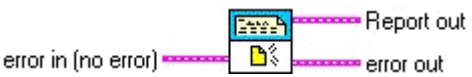
Get Report Settings

Retrieves information about the current font and text settings of a given report. The information is displayed on the front panel.



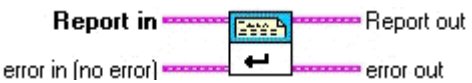
New Report

Creates a new report. You must use this VI to create a report if you do not use the Easy NI Report VI.



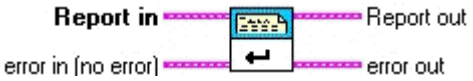
New Report Line

Starts a new line in the report to which you can append text or append another file.



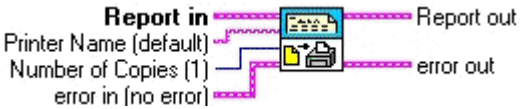
New Report Page

Adds a new page in the report to which you can append text or append another file.



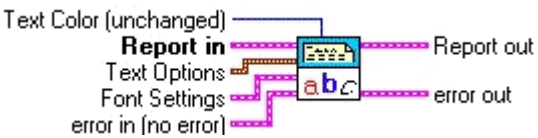
Print Report

Prints the report to a designated printer or to the default printer set up on your computer.



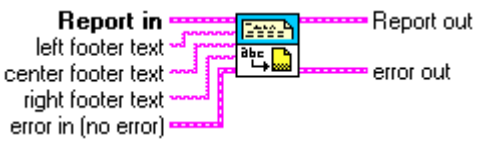
Set Report Font

Sets the font properties for the report, including those in the headers and footers. The available options include italic, bold, strikethrough, underline, color, font name, font size, character set and weight.



Set Report Footer Text

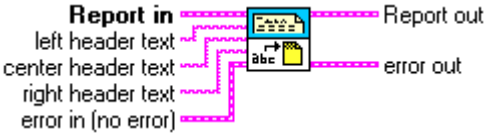
Sets the text for the report footer.



You might find that using tokens with this VI is helpful, particularly in the **Headers** and **Footers** parameters. For example, you might want to put a time stamp in the footer. See the *Tokens* topic, earlier in this chapter, for more information on some common tokens for this VI.

Set Report Header Text

Sets the text for the VI header.

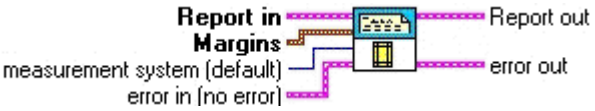


You might find that using tokens with this VI is helpful, particularly in the **Headers** and **Footers** parameters. For example, you might want to give the report a time stamp in the header. See the *Tokens* topic, earlier in this chapter, for more information on some common tokens for this VI.

Set Report Margins

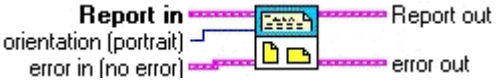
Sets the margins of the specified report.

Margins are defined by a cluster of four values: *left*, *right*, *top*, and *bottom* margins. Also, you must specify the measurement system (US, Metric, or Default) that determines your units of measurement. Selecting **US** gives you inches, selecting **Metric** gives you centimeters, and selecting **Default** gives you the units of the current measurement system configured on your computer.



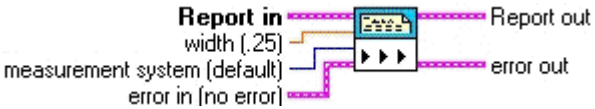
Set Report Orientation

Determines whether the report is printed in landscape or portrait orientation.



Set Report Tab Width

Sets the tab width in the report. To place a tab in your text, insert the token `<tab>` in the text string. Alternatively, you can use the Concatenate String function and insert a tab constant to build the text string.



Enhancements to DAQ

This section describes the data acquisition (DAQ) enhancements in LabVIEW 5.1.

DAQ Solution Wizard

You can now use the DAQ Solution Wizard even if you have no DAQ devices configured on your computer. However, without DAQ devices installed on your computer you cannot specify particular board options, and any opened solutions cannot be run until you install the appropriate DAQ device.

Support for NI-DAQ for Windows and Macintosh

LabVIEW 5.1 for Windows 95/NT ships with NI-DAQ 6.5. LabVIEW 5.1 for the Macintosh platform ships with an updated version of NI-DAQ 6.1. The DAQ Channel Wizard has been upgraded in NI-DAQ 6.5 and now is integrated in with the Measurement & Automation Explorer (the NI-DAQ configuration utility), which allows you to configure and test your National Instruments products from one common application.

New Syntax Element for Nonsequentially Scanned SCXI Module Channels

If you operate a module in parallel mode, you can specify an SCXI channel either by specifying the corresponding onboard channels or by using the *SCXI channel syntax*. This syntax is described in the *SCXI Channel Addressing* section of Chapter 20, *Special Programming Considerations for SCXI*, in the *LabVIEW Data Acquisition Basics Manual*.

If you are scanning modules, you can scan an arbitrary number of channels for each module using channel list elements that are components of the SCXI channel syntax. Previously, however, the channels of each module needed to be scanned in consecutive, ascending order.

Now, with LabVIEW 5.1, you can scan modules randomly, provided that the module is capable of random scanning. The new syntax appears below:

Channel List Element	Channel Specified
`OBz!SCx!MDy!(a,…,n)`	Channel `a` through `n` in the module in slot `y` of the chassis with ID `x` are multiplexed randomly onto onboard channel `z`.

For example, if you wanted to sequentially scan channels 2, 3, 4, and 5, you could use the following channel list element:

```
ob0!sc1!md2!(2,3,4,5)
```

You could also use the channel list element `ob0!sc1!md2!2:5`.

However, to scan the module's channels randomly, you could use the following channel list element:

```
ob0!sc1!md2!(5,1,3,5)
```

You could also use a colon (:) in the list to scan a series of channels sequentially, as in the following channel list element:

```
ob0!sc1!md2!(2,1,4,7:11,13,15)
```

In addition, you can use semicolons instead of commas within the parentheses and achieve the same scan:

```
ob0!sc1!md2!(2;1;4;7:11;13;15)
```

Enhancements to VISA

LabVIEW 5.1 has two new VISA functions as well as a new VISA palette that contains the interface-specific functions. You can reach the new palette by selecting **Functions»Instrument I/O»VISA»Interface Specific**.

This palette replaces the **Functions»Instrument I/O»VISA»VISA Serial** palette. In versions of LabVIEW previous to LabVIEW 5.1, the following

three VIs could be found on the **VISA Serial** palette but are now on the **Interface Specific** palette:

- VISA Set Serial Buffer Size
- VISA Flush Serial Buffer
- VISA Serial Break

In addition to the above existing functions, you can find the following two new functions on the **Interface Specific** palette:

- VISA GPIB Control REN
- VISA VXI Cmd or Query

The VISA GPIB Control REN function asserts or deasserts the GPIB Remote Enable interface line according to a specified mode. The VISA VXI Cmd or Query function sends a command or query, or receives a response to a query previously sent to the device.

You can find examples of the VISA VIs in the `Examples\Instr\visa.llb` directory.

Enhancements for Building and Distributing Applications

This section describes the new features for building and distributing applications now available in LabVIEW.

Note ***This functionality is available only in the LabVIEW Professional Development System or if you purchase the Application Builder Libraries.***

Building Executable Programs

In LabVIEW 5.1, the process for building an application has been streamlined. Previously, to do so you had to save your VIs to a library, then build an application using the **Build Application** dialog box. Further, to build an installer in Windows you had to use the **Create Distribution Kit** dialog box.

Now, in LabVIEW 5.1, you can use the **Build Application** dialog box to do all of these operations. You can configure the application to various settings within the tabs on the **Build Application** dialog box. After you define these settings, you can save them in a script so that you can easily rebuild the application.

Building an Application

Complete the following instructions to build an application in LabVIEW.

1. Select **Project»Build Application…**. The **Build Application** dialog box appears. The **Build Application** dialog box contains the following tabs: **Target**, **Source Files**, **VI Settings**, **App Settings**, and **Installer**. You can create a new build or load a build file that you created previously.
   - If you want to create a new application, proceed to Step 2.
   - If you already have a build file, click **Load…** and choose the `.bld` file to load. Then proceed to Step 6.
2. From the **Target** tab, specify the following information:
   - **Application name**—The name of the application you are creating. This file should have a `.exe` extension.
   - **Destination directory**—The path and name of the directory in which to create and save your new application.
   - **Support file directory**—The path and name of the directory in which to save any support files.
3. Under the **Build** section of the **Target** tab, choose one of the following two options:
   - **Single application containing all VIs**—Check this option to create a single application containing all of your VIs.
   - **Small application with external file for subVIs**—Check this option if you want to keep the main application small.
4. From the **Source Files** tab, you can configure the VIs that make up your application. You can specify top-level VIs, dynamically loaded VIs, and additional non-VI files (such as readme files). You can update the file list automatically as VIs are added or removed from your hierarchies. Depending on what kind of source files you want to add, complete the instructions below.
   a. If you want to add top-level VIs, click **Add Top Level VI…**. The **Open** dialog box appears where you can enter a file name, or browse to find the VIs you want to add. When you select a top-level VI, LabVIEW automatically includes all its subVIs and related files (such as menu files or DLLs).
   b. If you want to add dynamic VIs, click **Add Dynamic VI…**. If your VI dynamically calls any subVIs using the VI Server, LabVIEW cannot detect them automatically, so you must add them by using this option.

c. If you want to add support files, click **Add Support File…**. When you use this option, data files copy over to your application directory automatically. In addition to VI files (VIs, controls, menus, external subroutines, and so on), you can determine the set of DLLs referenced by your application. Because there are DLLs you might not want to redistribute, LabVIEW includes only those DLLs that are within the source hierarchy directories and the LabVIEW directory. If you want to include DLLs that are in the System directory, for example, you can include them as additional non-VI files manually.

d. If you want to remove a file from the list, click the file to highlight it and click **Remove File**.

5. Click **Save** to save the information you have entered. The **Save As** dialog box appears. Enter a file name with a `*.bld` extension to save the information you have entered into this dialog box.
6. Click **Build**. The **Build Status** dialog box appears.
7. After the build operation finishes, click **Done** to close the **Build Application** dialog box.

Customizing Application Features

1. If you want to customize some destination settings, select **Custom Destinations…** from the **Source Files** tab. The **Destination Settings** dialog appears, in which you can configure the following settings:
   - You can modify your destination directory.
   - **(Windows)** If you want to add a program item to your **Start** menu as part of an installer, select the **Create program item** checkbox and enter the name.
   - **(Windows)** If you are creating an installer, you can specify how you want to **Replace existing files**. Select **Never**, **Ask**, **If Newer**, or **Always**, depending on how or if you want to be prompted.
2. **(Windows** and **Macintosh)** From the **App Settings** tab, you can customize the features in your application. You can choose to specify the memory size for the Macintosh, or customize icons and ActiveX server features on Windows.
   a. **(Windows)** If you want to specify your own icon, click the **Custom icon** checkbox and designate the path to the icon.
   b. **(Windows)** If you want to enable the ActiveX server, click the **Enable ActiveX server** checkbox. Your application can then respond to requests from ActiveX clients. The functionality of the ActiveX server in your application is a subset of the LabVIEW

ActiveX server. When you build an application `myapp.exe`, an ActiveX type library `myapp.tlb` is also created along with the executable. The type library defines a createable class, *Application*, and a dispatch class, *Virtual Instrument*, and exports the properties and methods for these classes. You can find the Help for these properties and methods in `lvcomm.hlp` in the LabVIEW `Help` directory. When you distribute the application make sure the type library and the help file are located with the executable.

When you assign the name of the application to the server name, your application is uniquely identified in the system registry. Once you build the application, you should run it at least once to enable registry with the system. After the application is registered, ActiveX clients access the server objects using server names. For example, if you specify the server name as `myapp`, clients instantiate an application object using the `myapp.application`.

c. **(Macintosh)** Use the Memory Size control to specify the memory allocated to the application.

Modifying VIs as Part of the Build

Use the **VI Settings** tab to specify the modifications to your VIs that you want to make part of the application build. You can choose to enable or disable various window option and execution option VI Setup settings. These settings apply to the build process only and do not affect your original source VIs.

LabVIEW removes debugging code, block diagrams, and unnecessary front panels, making your application as small as it can be. The removal of front panels is a new feature with LabVIEW 5.1. LabVIEW can detect which panels are necessary in almost all cases. However, if you open a front panel dynamically using the VI Server, you must specify that the panel is needed using the **VI Settings** tab.

You can edit only a single row at a time. By default, all unnecessary panels are removed. You can override the default and include the panel by setting the **Remove Panel** option to **No**.

To change settings, select a VI so that it is highlighted in the list. Click **Edit Build Settings…**. The **Edit Build Settings** dialog box appears. For each setting you can choose **yes**, **no**, or **no change**. When you have made all the settings, click **Change**. Verify that all the settings are the way you want them for each VI in the build.

Note ***This completes the build application process on the UNIX and Macintosh platforms. The steps described in the following section apply to Windows only.***

Creating an Installer (Windows only)

1. From the **Installer** tab, click the **Create installer** checkbox. Verify the following sections of this tabbed page. If you create an installer, the installer is written to the directory that contains your application. The disk images are created in a *disks* subdirectory of the destination directory that you specified on the **Target** tab. This directory will contain a setup program as well as files named `data.001`, `data.002`, and so on. If you plan to put the disk images on floppy, it is best to copy the `setup` and `data.001` files to the first floppy, copy the `data.002` file to the second floppy, and so on.
   - Installation name
   - Start menu program group
   - Default installation directory
   - Installation language
   - Media size
   - Extra space on first disk (KB)

   The **Media size** item lets you specify how the file is to be segmented—for 720 KB, 1.2 MB, or 1.4 MB floppies. Even if you plan to distribute the files by CD, it is necessary to segment them. However, if you want to run the installer from a CD or from your drive, you can place all of the files in the same directory and run the setup program from that directory.

   The **Extra space on first disk (KB)** item lets you reserve space on the first disk. You might reserve space on the first disk if you want to put a readme file on the first floppy.
2. Click the **Advanced** button. The **Advanced Installer Settings** dialog appears.
   a. If you would like to create an uninstaller, click the **Create uninstaller** checkbox.
   b. If you would like to run a program after the installation, click the **Run executable after installation** checkbox and enter the executable and command line argument information.
3. Select the **Run executable after installation** item if you want to run a program after the installation completes. Additionally, you can use this item to run a program that finishes the installation. For example, you might write a DOS batch program or a C program that modifies a `.ini`

file or a `registry` file. Install the file as part of your installation and then run it afterwards to make the necessary modifications. The file that you run must be one of the files that you install.

4. If you choose to run an executable after the installation completes, you can use the **Command line arguments** to specify arguments passed to the program. In addition to specifying standard arguments, you can embed any of the following items in the command line argument string:

`%dest`	The application installation directory chosen by the user
`%src`	The directory that contains `setup.exe`
`%group`	The installation program group name
`%name`	The installation name

If any of these options are present at installation time, they are replaced with the proper values before the arguments are passed to the executable.

Run-Time Engine for the Application Builder for Windows

When you develop an executable program with LabVIEW for Windows and ship it to another computer, you must also include the LabVIEW Run-Time Engine. The computer on which the program runs must install this component using the LabVIEW Run-Time Engine Installer before the program executes.

If you distribute a program using Build Application, the Run-Time Engine is installed automatically.

This enhancement greatly reduces the size of the executable program.

☞ **Note** ***After the Run-Time Engine is properly installed on a machine, it can run any executable program developed in LabVIEW. You only have to include the Run-Time Engine with the first program sent to each computer.***

Enhancements to Networking

This section describes the networking enhancements in LabVIEW 5.1.

DataSocket VIs for Windows

DataSocket technology facilitates the exchange of data and information between an application and a number of different data sources and targets. These sources and targets include files and HTTP/FTP servers.

In LabVIEW 5.1, new VIs provide a simple yet intuitive way to access and use DataSocket technology within LabVIEW. You can create applications that share data among many different sources using a single Application Programming Interface (API).

The DataSocket VIs are available only for Windows platforms. For more information about the new DataSocket VIs, refer to the DataSocket VIs topic in the LabVIEW *Online Reference*. You can find examples of the DataSocket VIs in the `Examples\Comm\datasktx.llb` directory.

Internet/HTTP Services

Internet/HTTP services are available on all platforms and versions for LabVIEW 5.1, and give you capability to do the following:

- Publish HTML documents
- Generate HTML-readable images of a VI's front panel across the Web
- Generate animated versions of a VI's front panel on the Web
- Use basic access control to limit which VIs may be viewed and by whom

☞ **Note** ***If you want to control VIs from a browser, or you need to dynamically create HTML documents from your LabVIEW program, or you need more sophisticated access control, consider the Internet Toolkit, also available from National Instruments.***

The **Edit»Preferences** dialog box now includes three new list box options that you use to set up the built-in Web Server:

- **Web Server: Configuration**
- **Web Server: Browser Access**
- **Web Server: Visible VIs**

The Web Server

With the Web Server in LabVIEW, you can publish your VI front panels on the World Wide Web along with HTML documents.

The built-in Web Server is intended for users that need to make the panels of their running VIs visible on the Internet, but do not require sophisticated security features nor want to control VIs through the Internet.

The Web Server Configuration Dialog Box

You use the **Web Server Configuration** dialog box to set up the Web Server.

Access the **Web Server Configuration** dialog box by selecting **Edit»Preferences** and then **Web Server: Configuration** from the drop-down menu. Table 2-6 describes the available options.

Table 2-6. Web Server Configuration Dialog Box Options

Option	Description
Web Server Enabled	Enables the Web Server to publish front panel images and HTML documents. The default status is `Off`.
Root Directory	The directory that contains HTML documents published by the LabVIEW Web Server. The default path is `<LabVIEW>\www`.
HTTP Port	The TCP/IP port used to access the Web Server. The default port is `80`.
Timeout	How long (in seconds) the Web Server waits while reading a request before it times out. The default value is `60`.
Log File	The path to the data log file in which time-stamped information about connections is saved. The default path is `<LabVIEW>\www.log`.

The Web Server Browser Access Dialog Box

The **Web Server Browser Access** dialog box lists the remote computers on the Internet that are allowed to access the Web Server.

Access the **Web Server Browser Access** dialog box by selecting **Edit»Preferences** and selecting **Web Server: Browser Access** in the drop-down menu.

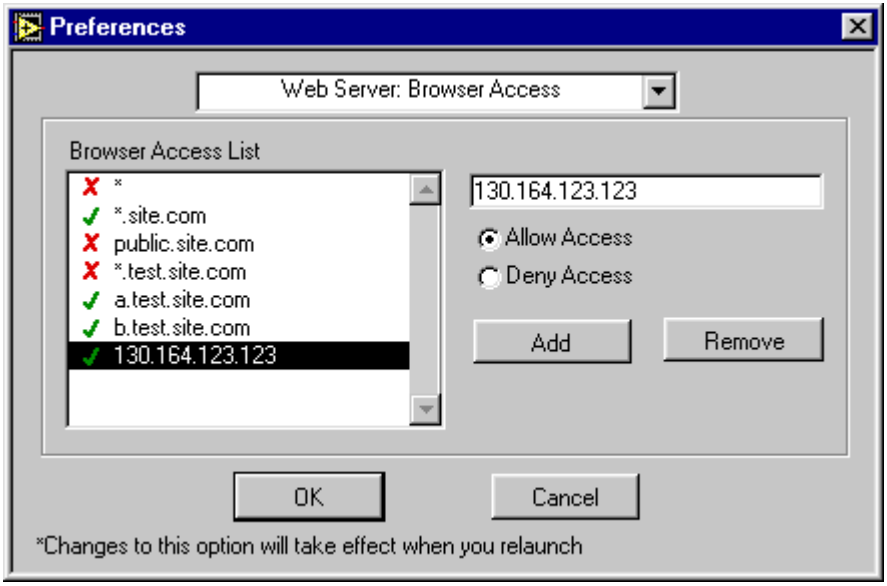


Figure 2-6. Web Server Browser Access Dialog Box

Table 2-7 describes the available options.

Table 2-7. Web Server Browser Access Dialog Box Options

Option	Description
Browser Access List	Lists computers or domains that have access to the Web Server. Click and drag an entry in this list to move it in the access list. If an entry permits access to the Web Server from an address, a check mark appears next to its name. If an entry denies access to the Web Server, an **X** appears next to its name. If no symbol appears next to the entry, the syntax for the entry is incorrect.
Text box	Changes or adds an entry to the list. To add an entry, click the **Add** button and type in this box. To change an existing entry, select it from the **Access List** and edit it in this box.
Allow Access and **Deny Access**	Determines whether the current entry has access to the Web Server. Click the **Allow Access** radio button to grant a computer or a domain access to the Web Server. Click the **Deny Access** radio button to deny a computer or domain access to the Web Server.
Add	Adds a new entry to the **Access List** following the item currently highlighted in the **Access List**.
Remove	Removes the highlighted entry from the **Access List**.

When a browser attempts to connect to the Web Server, the server examines the entries in the **Browser Access List** to determine whether the computer is permitted access. If an entry in the list matches the computer's address, the server either permits or denies access, based on how you set up the entry. If a subsequent entry also matches the computer's address, that permission is used in place of the previous permission.

For example, if you give `a.test.site.com` and `b.test.site.com` access, but do not extend access to *all* addresses ending in `.test.site.com` (where the * wildcard indicates all), the two computers still have access. If no entry matches the client address, access is denied. (See Table 2-8 for more information on the * wildcard and permitting matching access entries).

To specify an Internet host address, enter its domain name or IP address. You can use the * wildcard when specifying Internet host addresses. For example, you can specify all hosts within the domain `domain.com` with the entry `*.domain.com`. You can specify all hosts in the subnet whose first two octets are `130.164` with the entry `130.164.*`. The entry `*` matches all addresses.

Table 2-8 shows examples of TCP/IP Access List entries.

Table 2-8. Examples of Access List Entries

Access String	Matches
`*`	All hosts
`test.site.com`	The host whose domain name is `test.site.com`
`*.site.com`	All hosts whose domain names end with `.site.com`
`130.164.123.123`	The host with the IP address 130.164.123.123
`130.164.123.*`	All hosts whose IP addresses start with 130.164.123

In the *Web Server Browser Access Dialog Box* shown earlier in Figure 2-6, all hosts in the `site.com` domain have access to the server, with the exception of all hosts in the `test.site.com` domain. Additionally, the hosts `a.test.site.com`, `b.test.site.com`, and `130.164.123.123` also have access to the server. The host `public.site.com` does not have access, even though it is in the `site.com` domain.

By default, all hosts have access to the Web Server.

Note ***If the Web Server runs on a system that does not have access to a DNS server, do not use domain name entries in the Access List. Requests to resolve the domain name or an IP address fail, slowing down the system. For performance reasons, place frequently matched entries toward the bottom of the Access List.***

The Web Server Visible VIs Dialog Box

Because the Web Server publishes front panels to the Web, it is best to specify which VI front panels you allow to be published. Through the **Web Server: Visible VIs** dialog box, you specify which front panels can be published to the Web.

You access the **Web Server Visible VIs** dialog box by selecting **Edit»Preferences** and selecting **Web Server: Visible VIs** in the drop-down menu.

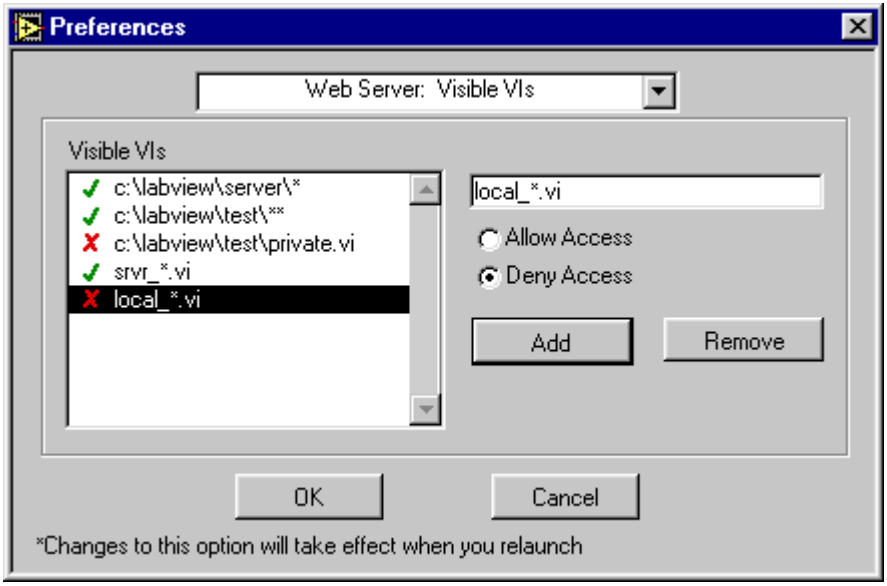


Figure 2-7. Web Server Visible VIs Dialog Box

Table 2-9 describes the available options.

Table 2-9. Web Server Visible VIs Dialog Box Options

Option	Description
Visible VIs	Specifies the VIs or groups of VIs whose front panels may be published by the Web Server. Click and drag an entry in this list to move it in the **Visible VIs** list. If an entry permits a VI or group of VIs to be seen, a check mark appears next to its name. If an entry denies such access, an **X** appears next to its name. If no symbol appears next to the entry, the entry's syntax is incorrect.
Text box	Changes or adds an entry to the list. To add an entry, click the **Add** button and type in this box. To change an existing entry, select it from the **Visible VIs** list and edit it in this box.
Allow Access and **Deny Access**	Determines whether the front panel of the current entry can be published by the Web Server. Click the **Allow Access** radio button if you want to allow the front panel of the VI or group of VIs to be published by the Web Server. Click the **Deny Access** radio button if you do not want to allow the front panel of the VI or group of VIs to be published by the Web Server.
Add	Adds a new entry to the **Visible VIs** list after the current item highlighted.
Remove	Removes the highlighted entry from the **Visible VIs** list.

Each entry in the **Visible VIs** list describes a VI name or a VI path and might contain wildcard characters. Entries that contain path separators are compared against VI paths, while entries that do not contain path separators are compared against VI names only.

When a web browser attempts to obtain a VI front panel image, the server examines the **Visible VIs** list to determine if it should grant access to the requested VI's image. If an entry in the list matches the requested VI, the Web Server either permits or denies access to that VI's image, based on

how that entry is set up. If a subsequent entry also matches the VI, its access permission is used in place of the previous permission. If there is no VI in the list that matches the requested VI, access to the VI's image is denied.

You can use wildcard characters in the **Visible VIs** list so an entry in the list matches more than one VI. Use the wildcard characters shown in Table 2-10.

Table 2-10. Wildcard Characters in Visible VIs List

Wildcard	Action
`'?'`	Matches exactly one arbitrary character, except for the path separator
`'*'`	Matches zero or more arbitrary characters, except for the path separator
`'**'`	Matches zero or more arbitrary characters, including the path separator

If you want to match a VI with a name that contains a wildcard character, you must escape that character using `'\'` (Macintosh and UNIX), or `'`` '` (Windows).

Table 2-11 shows examples of Visible VI list entries. The examples use UNIX path separators.

Table 2-11. Examples of Visible VI List Entries

VI Access String	Matches
`*`	All VIs
`/usr/labview/*`	All VIs in the directory `/usr/labview/`.
`/usr/labview/**`	All VIs in the directory `/usr/labview/` and any of its sub-directories
`Test.vi`	Any VI named `Test.vi`
`OK\?`	Any VI with the name `OK?`

In the *Web Server Visible VIs Dialog Box* shown earlier in Figure 2-7, all VIs in the `c:\labview\server` directory have front panels that you can see on the web. All VIs in the `c:\labview\test` directory and all its sub-directories are exported as well, with the exception of the VI `c:\labview\test\private.vi`. Additionally, any VI that begins with

the string `srvr_` and ends with the string `.vi` is exported. No VI that begins with the string `local_` and ends with the string `.vi` is exported, even if it is located within the `c:\labview\server` directory.

By default, the front panel image of all VIs are visible.

Configuring the Web Server

Complete the following steps to configure the Web Server.

1. Select **Edit»Preferences** and choose **Web Server: Configuration** in the drop-down menu.

   See *The Web Server Configuration Dialog Box* section earlier in this chapter for more information on this dialog box.

2. Enter a **Root Directory**, which is the directory where the Web Server's HTML files are located. The default path is `<LabVIEW>\www`.

3. In **HTTP Port**, specify the TCP/IP port the server uses.

   The default port for HTTP is 80. You might specify a different port if another HTTP Server already uses port 80 on your machine or if you are on a system where you do not have permission to use reserved ports.

   If you use a non-default port, such as 8000, you must specify it on URLs that refer to your server, as shown in this example: `http://hostname:8000/index.htm`.

4. In **Timeout** specify the number of seconds the Web Server waits while reading a request before the server times out. The default value is `60`.

5. Specify a **Log File**, which is the path of the file where information about web connections is saved. The default path is `<LabVIEW>\www.log`.

Running the Web Server

You must run the Web Server in order to publish VI front panel images on the World Wide Web.

To run the Web Server, check **Web Server Enabled** in the **Preferences»Web Server: Configuration** dialog box.

Publishing Front Panel Images on the Web

Complete the following steps to publish a VI front panel image on the World Wide Web.

1. Run the Web Server by checking **Enable Web Server** in the **Preferences»Web Server: Configuration** dialog box.
2. Launch the VI(s) you want to publish to the web so they are in your system's memory.

Retrieving a Static Image of a VI Front Panel

☞ **Note** ***To retrieve an HTML-Readable Image of a VI front panel, the VI must be in memory.***

Complete the following steps to retrieve an HTML-readable image of a front panel.

1. Open a Web browser.
2. Type in a URL that includes the address of the LabVIEW Web Server, the command for a static image (`.snap`), a question mark (?) to separate the URL from the parameters, and the name of the VI. The URL takes the following form:

   `http://web.server.addr/.snap?VI_Name`

You must encode the VI Name according to URL naming rules. Replace special characters with their hexadecimal value preceded by a percent (%) sign and replace spaces with a plus (+) sign.

For example, you can write the URL for the static image of the `Test Example.vi` as follows:

`http://web.server.addr/.snap?Test+Example.vi`

Following the VI name, you can add parameters that specify attributes for the image. See *Static Front Panel Image (.snap URL)* in the *What URLs Can I Use with My Front Panel Images?* section later in this chapter for more information on the characters you can enter in URLs to retrieve images.

Viewing an Animated Version of a Front Panel

Complete the following steps to view an animated version of a front panel.

1. Open a Web browser.
2. Type in a URL that includes the address of the LabVIEW Web Server, the command for an animated image (`.monitor`), a question mark (`?`) to separate the URL from the parameters, and the name of the VI. The URL takes the following form:

   `http://web.server.addr/.monitor?VI_Name`

You must encode the VI Name according to URL naming rules. Replace special characters with their hexadecimal value preceded by a percent (%) sign and replace spaces with a plus (+) sign.

With Netscape Navigator browsers, the Web Server uses the server-push method to implement animations of front panel images. During a server push, the server maintains an open connection and sends a new image after a predefined period of time. With other browsers, the Web Server uses the client-pull method to implement animation. During a client-pull animation, the browser intermittently sends a request for each image.

Following the VI name, you can add parameters that specify attributes for the image animation. See *Animated Front Panel Image (.monitor URL)* in the *What URLs Can I Use with My Front Panel Images?* section later in this chapter for more information on the characters you can enter in URLs to retrieve images.

Determining Which Front Panels are Visible

To make a VI's front panel visible across the Web.

1. Open the **Web Server Visible VIs** dialog box by selecting **Edit»Preferences** and selecting **Web Server: Visible VIs** in the drop-down menu.
2. Choose **Add**.
3. To deny a specific VI's front panel from being viewed, type its path in the **Text box** and choose **Deny Access**. To allow a specific VI's front panel to be viewed, types its path in the **Text box** and choose **Allow access**.

Note ***You also can allow or disallow the viewing of whole groups of VI front panels by using wildcards. See the section The Web Server Visible VIs Dialog Box, earlier in this chapter for more information on wildcards.***

What URLs Can I Use with My Front Panel Images?

With the Web Server, you can publish images of your VI front panels on the World Wide Web. You do not need to modify the VIs to display their front panels.

Front Panel Image Formats

The Web Server can generate images of VI front panels in the Joint Photographic Experts Group (JPEG) and Portable Network Graphics (PNG) image formats.

The JPEG image format is a public domain image format that all current browsers support. It has been developed for the distribution of real-life images and photographs and uses a lossy compression algorithm to reduce the memory size of an image. When you use JPEG on images that contain lines and text, such as front panels, the resulting image often displays artifacts of the compression, such as fuzzy text or stray color pixels.

The PNG format is a recent public domain image format. The compression algorithm in this format is lossless, which produces PNG images exactly like the original images. PNG is designed to be the successor of the Graphics Interchange Format (GIF), which also uses lossless compression. PNG is an open standard that you also can use on true-color images. Internet Explorer 4.0.1 and Netscape Navigator 4.0.4 support the PNG format. Older browsers require a plug-in or an external application to view PNG images.

Static Front Panel Image (.snap URL)

The .snap URL signals the server to return a static image of the front panel of a VI currently in memory. The query parameters in the URL specify the VI name and the attributes of the image.

You must open the front panel of the VI to take snapshots for static images because closed front panels do not update the images of controls when the value changes.

The syntax to use in the URL for static front panel images is as follows (parameters in brackets [ ] are optional):

.snap?VI_Name

[**&type=type**]

[**&depth=depth**]

[**&quality=quality**]

[**&compression=compression**]

[**&refresh=refresh**]

[**&full=full**]

VI_Name is the name of the returned VI front panel. You must encode the VI name according to HTTP conventions. Replace special characters with *%xx*, where *xx* is the hexadecimal value of the character.

Type is the returned image type, either JPEG or PNG. If no **type** is specified, the default type is used.

Depth is the depth of the returned image. **Depth** can be 1, 4, 8, or 24 bits. If no **depth** is specified, the default depth is used.

Quality is the image quality and memory size of the JPEG front panel image. **Quality** can be between 0 and 100. If no **quality** is specified, the default quality is used.

Compression is the compression level used for compressing PNG images. **Compression** can be between 0 and 7. If no **compression** is specified, the default PNG compression is used.

Refresh is the maximum age of a cached image. If a cached image is older than refresh seconds, a new image is generated.

Full specifies whether to return the image of all controls or just the part visible in the window. Set **full** to **on** to indicate all controls and **off** to indicate the window content. If no **full** is specified, the image of the visible front panel in the window is returned.

The following are some examples on how you would use this syntax:

- To return the front panel image of the VI `My VI.vi` from the computer `foo` using the default image **type**, **depth**, and **quality**, use the following code:

  ```
  http://foo/.snap?My%20VI.vi
  ```

- To return the front panel image of the VI `Test 1.vi` from the computer `foo` using image **depth**=24 and image **type**=PNG, use the following code:

  ```
  http://foo/.snap?Test%201.vi&depth=24&type=png
  ```

- To embed the image of the VI `Example.vi`, in an HTML document on the same system, use the following code:

  ```
  <IMG SRC="/.snap?Example.vi">
  ```

- To embed the image of the VI `Example.vi` running on the computer `foo` in any HTML document, use the following code:

  ```
  <IMG SRC="http://foo/.snap?Example.vi">
  ```

Animated Front Panel Image (.monitor URL)

The .monitor URL signals the server to return an animated image of the front panel of a VI currently in memory. The query parameters in the URL specify the VI name, attributes of the animation, and attributes of the image.

For example, you can write the URL for the animated image of the VI `Test Example.vi`, which updates once every two seconds for three minutes, as follows:

```
http://web.server.addr/.monitor?Test+Example.vi&refresh=2&lifespan=180
```

For Netscape Navigator browsers, the server uses the server-push method of animation. The server accomplishes this animation by taking subsequent snapshots of the front panel image and sending them to the client. With other browsers, the server uses the client-pull method of animation.

You must open the front panel of the VI to take snapshots for animated images because closed front panels do not update the images of controls when the value changes.

The syntax to use in the URL for static front panel images is as follows (parameters in brackets [ ] are optional):

.monitor?VI_Name

[**&type=type**]

[**&depth=depth**]

[**&quality=quality**]

[**&compression=compression**]

[**&refresh=refresh**]

[**&full=full**]

[**&lifespan=lifespan**]

VI_Name is the name of the returned VI front panel. You must encode the VI name according to HTTP conventions. Replace special characters with %*xx*, where *xx* is the hexadecimal value of the character.

Type is the returned image type, either JPEG or PNG. If no **type** is specified, the default type is used.

Depth is the depth of the returned image. **Depth** can be 1, 4, 8, or 24 bits. If no **depth** is specified, the default depth is used.

Quality is the image quality and memory size of the JPEG front panel image. **Quality** can be between 0 and 100. If no **quality** is specified, the default quality is used.

Compression is the compression level used for compressing PNG images. **Compression** can be between 0 and 7. If no **compression** is specified, the default PNG compression is used.

Refresh is number of seconds between each succeeding image.

Full specifies whether to return the image of all controls or just the part visible in the window. Set **full** to **on** to indicate all controls and **off** to indicate the window content. If no **full** is specified, the image of the visible front panel in the window is returned.

Lifespan is the number of seconds the front panel animation lasts. Setting `Lifespan=0` implies that the animation continues until the browser cancels it. If no **lifespan** is specified, the default lifespan is used.

☞ **Note** ***When using client-pull animations, the lifespan is ignored.***

The following are some examples on how you would use this syntax:

- To generate an animated front panel image of the VI `My VI.vi` from the computer `foo` using the default image **type**, **depth**, and **quality**, use the following code:

  ```
  http://foo/.monitor?My%20VI.vi
  ```

- To generate a 60-second animation of the front panel image of the VI `Test 1.vi` from the computer `foo` using the default image **type** and **quality** but using `refresh=5`, use the following code:

  ```
  http://foo/.monitor?Test%201.vi&refresh=5&lifespan=60
  ```

- To embed the image of the VI `Example.vi` in an HTML document on the same system, use the following code:

  ```
  <IMG SRC="/.monitor?Example.vi">
  ```

- To embed the image of the VI `Example.vi` running on the computer `foo` in any HTML document, use the following code:

  ```
  <IMG SRC="http://foo/.monitor?Example.vi">
  ```

Enhancements to Examples and Activities

You can run example VIs to help you get started with LabVIEW. To generate or find examples similar to your application, refer to the Solution Wizards (on Windows and PCI Macintosh only) or to the Search Examples online help file (Windows only), which you can access from the LabVIEW startup dialog box.

In addition to the example VIs, a good way to get started with LabVIEW is to complete the activities available to you in the *LabVIEW User Manual* and in the LabVIEW *Online Reference*.

☞ **Note** ***The LabVIEW Online Reference has the most up-to-date information regarding paths to VIs, functions, and controls.***

A

Manual Clarifications and Additions

This appendix clarifies and corrects information in the LabVIEW manual set. Because these manuals were not revised for the version 5.1 release of LabVIEW, this appendix contains information relevant only to the previous release.

Enhancements to LabVIEW 5.1 have rendered some information in these manuals incorrect, particularly illustrations of palettes and navigation paths to functions and controls, including those in activities. See Chapter 2, *New Features in LabVIEW 5.1*, for updated information and descriptions of new features.

Multithreading

Color of Code Interface and Call Library Function Nodes—The color of a code interface node (CIN) or Call Library Function node on a block diagram changes depending on whether LabVIEW considers it reentrant. If LabVIEW considers a CIN or Call Library Function node reentrant, LabVIEW assigns it the current primitive color (the default is pale yellow). If a CIN or Call Library Function node is not considered reentrant, its color is orange. This color designation exists on all platforms, even if the platform itself is not threaded.

ActiveX

ole_lv5container.dll—The ActiveX Container uses a DLL named `ole_lv5container.dll`, which is located in the `resource` directory. If you build an application that includes ActiveX controls and move it to another machine, you must install this file in the same directory as the built application or in the `System` directory. In the LabVIEW documentation, references to `ole_container.dll` should be `ole_lv5container.dll`.

Data Format—The compatibility VIs for the LabVIEW 4.*x* Automation functions require that you pass flattened data in the LabVIEW 4.*x* format. LabVIEW 5.*x* loads your LabVIEW 4.*x* VIs and automatically selects the **Convert 4.x Data** option for the Flatten To String and Unflatten From String functions, which are found in the **Advanced»Data Manipulations** palette.

Instrumentation

Signal Generator by Duration VI—The Signal Generator by Duration VI has been added to the **Signal Processing»Signal Generation** palette. This VI generates a signal with a shape given by the waveform type: sine, cosine, triangle, square, sawtooth, increasing ramp, or decreasing ramp.

CVI Function Panel Converter Changes—The improved CVI Function Panel Converter creates hierarchical text menus so you can find functions quickly. Two new options have been added to the CVI Function Panel Converter. These options are ON by default.

- **Map ViSession type to VISA Session RefNum**—This option specifies that instrument session numbers of type ViSession in the CVI Function Panel are converted to LabVIEW VISA RefNums in the resulting VI. Functions that contain the string `_init` in their name automatically register with the VISA refnum; functions that contain `_close` in their name automatically close the VISA refnum.
- **Create instr.lib menu mirroring CVI Class Hierarchy**—This option specifies that when converting a Function Panel file, a palette menu for the instrument is created in the **Instrument Drivers** menu. This menu is organized hierarchically according to the Function Panel Tree in the `.fp` file.

General Interface Features

Icon and Text Palettes

You can display palettes in three modes: Standard, All Icons, or All Text. Choose the palette display mode in the **Edit»Select Palette Set»Display Style** submenu.

In All Text mode, you right-click to access the **Controls** or **Functions** palette. These text palettes contain the names of options. Items in text palettes are organized in the same order as in the icon palettes when you read the icon palette row by row, left to right. Empty spaces in the icon

palettes are skipped. Unlike icon palettes, you cannot tack down text palettes or subpalettes. In All Text mode, LabVIEW displays text palettes in the **Project** menu and the Find dialog box. In Standard or All Icons mode, LabVIEW displays icon palettes.

Standard is the default mode. In Standard mode, all palettes default to icon palettes, but you can edit individual palettes to display them as text palettes.

When you edit a palette by selecting **Edit»Edit Control & Function Palettes…**, LabVIEW displays the palettes in All Icons mode. You cannot edit palettes in the other modes because they do not contain as much information (icon palettes have both icons and two-dimensional layout, while text palettes do not). To specify the mode for the Functions or Controls palette, right-click on the palette—but not on a subpalette icon—and select either **Icons** or **Text** from the **Standard Menu View** submenu. The mode you select affects only the menu you are editing.

File Manager Tool

The File Manager tool, which you access by choosing **Project» File Manager…**, simplifies copying, renaming, and deleting files within VI libraries (LLBs). You also can use this tool to create new LLBs and directories and convert LLBs to and from directories.

To avoid performing a file operation on a VI already in memory, close all VIs that might be affected before using this tool.

In the File Manager dialog box, shown in the following figure, you can view two locations (directory or LLB) simultaneously. When you select a file, you can copy, rename, or delete it using the corresponding buttons between the two lists. Click **New…** to create a new directory or LLB.

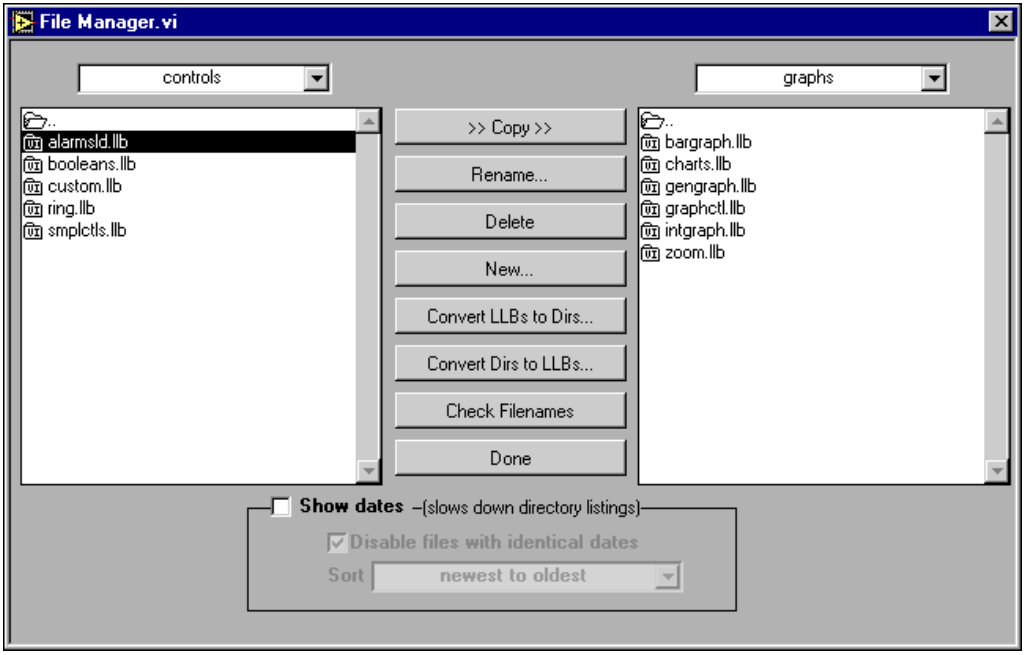


Figure A-1. File Manager Tool Dialog Box

If you select an LLB, you can click **Convert LLBs to Dirs** to convert it to a directory. If you select a directory and click this button, the tool scans for all LLBs within that directory and gives you the option to convert them to directories. The new directory is created in the same location as the original LLB.

If you assign the new directory a name that differs from that of the original LLB, LabVIEW searches for the files that were within the LLB when calling a VI (even when the name is the same minus the `.llb` extension). When you convert an LLB to a directory, you have the option to back up the LLB (the `.llb` extension changes to `.llx`).

To convert a directory to an LLB, select a directory and click **Convert Dirs to LLBs**.

Click **Check Filenames** to scan a directory or VI library for platform-dependent filenames. The tool scans all filenames for invalid characters (`:`, `\`, `/`, `?`, `*`, `<`, `>`, or `|`) and verifies filenames to be 31 characters or less (a limitation on the Macintosh). The **Check Filenames** option also scans files within LLBs. These files are portable, even if their names

contain characters that are invalid on some platforms. By scanning within LLBs, this tool helps you detect potential problems if you move your files out of VI libraries.

Use the **Show dates** option at the bottom of the dialog box to display file modification dates next to each file. You can choose to sort the files alphabetically or by date and disable files with the same name and date in both directory listings. Use this technique when comparing two directories to determine whether any files have changed.

Other General Interface Features

Dragging and Dropping VI Icons—LabVIEW 5.0 simplified the creation of VI icons. By dragging an image file and dropping it onto the VI icon in the upper-right corner of a front panel, a 32-by-32 version of the image replaces the existing icon.

You can drag a VI icon from the icon pane in the upper-right corner to a block diagram to instantly create a subVI call. By pressing <Shift> while dragging the VI icon, you automatically wire the non-default values of the controls as constants for the subVI.

If the subVI already appears in a block diagram, pressing <Shift> while dragging onto the existing call updates the attached constants. A control at its default value discards the constant attached to the subVI, and an input wired to anything other than a constant is unaffected.

When you press <Shift> while double-clicking a subVI icon to open the subVI front panel, LabVIEW loads the values of the constants wired to the subVI into the front panel controls. All unwired controls retain the default values.

You also can use the drag-and-drop technique for global variables and custom controls. Additionally, you can drag a VI icon into a VI refnum on a front panel control to load VIs into memory dynamically, which is part of the VI Server functionality.

Print to RTF/HTML Feature—The Print to RTF/HTML feature can export graphics in uncompressed graphics interchange format (GIF). To use this feature, select **File»Print Documentation**, and select **RTF File** or **HTML File** from the **Destination** pull-down menu.

Configuration File VIs—The Configuration File VIs, which you can access from the **Functions»File I/O»Configuration File VIs** palette, provide tools for reading from and writing to a platform-independent configuration file similar in format to a Windows initialization (`.ini`) file.

Macintosh Open Transport Support—LabVIEW 5.*x* supports Open Transport on Power Macintosh machines. Open Transport is a PowerPC-native networking driver.

New Preferences Options—LabVIEW 5.*x* adds the following two options in the **Miscellaneous** view of the **Edit»Preferences…** dialog box:

- **Automatically close VISA sessions**—Use this option to specify that VISA sessions, like file refnums, close automatically when the top-level VI goes idle. The default is ON, which closes VISA sessions automatically.
- **Treat read-only VI as locked**—Using this option, you can choose whether to treat read-only VIs as locked. You cannot edit locked VIs, but you can re-compile and execute them. By default the option is not selected so that read-only VIs appear normally. However, you cannot save the VI to the same location (the read-only file) unless you change the file permissions outside LabVIEW. This behavior is consistent with the behavior in previous versions of LabVIEW. When using the VI Server, the read-only status of files is ignored except when saving. This option is designed primarily to support the source-code control of the Professional G Developers Toolkit.

Execution System Selection—The default preferred execution system for a VI is **same as caller**. This setting allows the VI to run in the same execution system in which caller is running when the subVI call to the VI is made. The **same as caller** setting has the lowest run-time overhead. When you set a VI to **same as caller** and you run it at the top level, it runs in the **standard** execution system at its selected priority.

Icon Editor—The **Undo** button has been removed from the Icon Editor, but you can undo an action by selecting **Edit»Undo** or `<Ctrl-Z>`.

Offscreen Updates Default Value—The default value for offscreen updates is now ON instead of OFF.

Support for Template VIs and Controls

You can save commonly used VIs and controls as templates. To create a template VI, save a VI with a `.vit` extension (or `.ctt` extension for typedefs). When you open a template VI or control, the new file you create is named automatically using your template name and a number corresponding to the number of times it has been opened. When you finish editing the VI and save it, LabVIEW prompts you to enter a new name for the file.

To modify a template, open it, make your changes, then save over the `.vit` (or `.ctt`) file that you originally created.

(Macintosh) You also can use the **Stationery Pad** checkbox of the **Get Info** dialog box in the Finder to change a VI to a template.

Adding VIs to the Project and Help Menus

You can add VIs to the **Project** and **Help** menus by placing them inside the `Project` or `Help` directories in the `LabVIEW` directory. You can use this technique to provide quick access to VIs that act as tools in your system. National Instruments uses this feature to make the Tech Support VIs accessible from the **Help** menu. Also, if you have the Application Builder libraries installed, you can see a **Build Application…** option in the **Project** menu.

Any VI placed at the top level of the `Project` or `Help` directory is appended directly to the corresponding menu. If you create a subdirectory, a submenu is appended.

Allocation of Threads on Concurrent PowerMAX and Solaris 2

On Concurrent PowerMAX and Solaris 2, LabVIEW allocates threads as described below.

If LabVIEW has permission to increase its Light Weight Process (LWP) priorities from the default, it binds all created threads to LWPs.

- Profiling is very accurate because each thread is bound to a LWP and the kernel monitors the execution timing of LWPs.

- The LabVIEW priority system is reflected in the way the kernel runs LWPs. Higher-priority execution threads (LWPs) take over the system, not allowing lower-priority system tasks to execute.
- Switching between threads might require more time because LabVIEW runs through the system scheduler.

(PowerMAX) LabVIEW for Concurrent PowerMAX always binds threads to LWPs. Unless you have permission to adjust priorities to LWPs, LabVIEW will stop, indicating that it does not have enough permission to run. The permissions you need depends on the scheduler you are using. If you are using the time-shared scheduler—the default—you need the `P_TSHAR` privilege. If you are able to use the real-time scheduler, you do not need any additional privileges.

(Solaris 2) If LabVIEW for Solaris cannot increase its LWP priorities from the default, it creates a LWP per thread, but leaves the threads and LWPs unbound so the created threads have a pool of LWPs on which to run. The typical user does not have permission to raise LWP priorities. If LabVIEW threads are not bound to LWPs:

- Profiling strictly uses wall-clock time. Threads might switch LWPs dynamically without kernel knowledge, so LabVIEW cannot use LWP timing statistics.
- The LabVIEW priority system only has an effect internal to LabVIEW. The system treats all the LabVIEW LWPs as another process to schedule at the same priority as any other task in the system.
- Context switching between threads might be faster because it does not involve the kernel scheduler.

The About LabVIEW dialog box, which you can view by choosing **Help»About LabVIEW…**, indicates how LabVIEW currently allocates threads.

Clarifications to the LabVIEW User Manual

The following clarifications pertain to the *LabVIEW User Manual*:

- In Chapter 2, *Creating VIs*, the text and an illustration in Activity 2-3, *Create an Icon Connector*, refer to an **Undo** button in the Icon Editor. The **Undo** button has been removed, but you can undo an action by selecting **Edit»Undo** or <Ctrl-Z>.

- In Chapter 6, *Strings and File I/O*, the block diagram in Activity 6-3, *String Subsets and Number Extraction*, shows the From Exponential/Fract/Eng function. The block diagram should show the Scan From String function, as described in the text.
- In Chapter 7, *Getting Started with a LabVIEW Instrument Driver*, the *Interactively Testing Component VIs* section describes how to access open VISA sessions. On the pop-up menu of a VISA session control, if **Open Sessions...** is always grayed out, make sure that the **Automatically Close VISA Sessions** option in **Edit»Preferences»Miscellaneous** is unchecked.
- In Chapter 15, *Spectrum Analysis and Measurement*, the pathname for library that includes the THD Example VI in Activity 15-3, *Calculate Harmonic Distortion*, should be `examples\analysis\measure\measxmpl.ll`.
- **(Windows 95)** In the *Using NetDDE* section of Chapter 23, *Using DDE*, the manual refers to `REGEDIT` and `REDEGIT` executables. The correct name is `REGEDIT`.
- In Chapter 25, *Program-to-Program Communication*, the *PPC Client Example* section refers to the PPC Open Connection, PPC Open Session, PPC Close Session, and PPC Close Connection VIs. These should be the PPC Open Port, PPC Start Session, PPC End Session, and PPC Close Port VIs, respectively. The *PPC Server Example* section refers to the PPC Close Session VI, which should be the PPC End Session VI.

VISA Error Codes

The following table lists numeric VISA error codes that were not included in the printed documentation.

Error Code	Error Name	Description
1073676443	VI_SUCCESS_SYNC	Operation completed successfully, but the operation was actually synchronous rather than asynchronous.
1073676442	VI_SUCCESS_NESTED_EXCLUSIVE	Operation completed successfully, and this session has nested exclusive locks.
1073676441	VI_SUCCESS_NESTED_SHARED	Operation completed successfully, and this session has nested shared locks.
1073676440	VI_SUCCESS_NCHAIN	Event handled successfully. Do not invoke any other handlers on this session for this event.
1073676424	VI_WARN_NSUP_BUF	The specified I/O buffer is not supported.

Error Code	Error Name	Description
1073676421	VI_WARN_UNKNOWN_STATUS	The status code passed to the operation could not be interpreted.
1073676420	VI_WARN_NSUP_ATTR_STATE	Although the specified state of the attribute is valid, it is not supported by this resource implementation.
1073676418	VI_WARN_NULL_OBJECT	The specified object reference is uninitialized.
1073676416	VI_SUCCESS_QUEUE_NEMPTY	Wait terminated successfully on receipt of an event notification. There is at least one more event occurrence of the type specified by inEventType available for this session.
1073676413	VI_SUCCESS_DEV_NPRESENT	Session opened successfully, but the device at the specified address is not responding.
1073676407	VI_WARN_CONFIG_NLOADED	The specified configuration either does not exist or could not be loaded. VISA-specified defaults will be used.
1073676294	VI_SUCCESS_MAX_CNT	The number of bytes transferred is equal to the input count.
1073676293	VI_SUCCESS_TERM_CHAR	The specified termination character was read.
1073676292	VI_SUCCESS_QUEUE_EMPTY	Operation completed successfully, but queue was already empty.
1073676291	VI_SUCCESS_EVENT_DIS	Specified event is already disabled for at least one of the specified mechanisms.
1073676290	VI_SUCCESS_EVENT_EN	Specified event is already enabled for at least one of the specified mechanisms.
-1073807202	VI_ERROR_LIBRARY_NFOUND	A code library required by VISA could not be located or loaded.
-1073807204	VI_ERROR_SESN_NLOCKED	The current session did not have a lock on the resource.
-1073807215	VI_ERROR_INV_MODE	Invalid mode specified.
-1073807229	VI_ERROR_INV_LENGTH	Invalid length specified.
-1073807240	VI_ERROR_INV_PARAMETER	The value of some parameter (which parameter is not known) is invalid.
-1073807246	VI_ERROR_RSRC_BUSY	The resource is valid, but VISA cannot currently access it.
-1073807247	VI_ERROR_USER_BUF	A specified user buffer is not valid or cannot be accessed for the required size.
-1073807248	VI_ERROR_NSUP_ALIGN_OFFSET	The specified offset is not properly aligned for the access width of the operation.

Error Code	Error Name	Description
-1073807252	VI_ERROR_ASRL_OVERRUN	An overrun error occurred during transfer. A character was not read from the hardware before the next character arrived.
-1073807253	VI_ERROR_ASRL_FRAMING	A framing error occurred during transfer.
-1073807254	VI_ERROR_ASRL_PARITY	A parity error occurred during transfer.
-1073807263	VI_ERROR_NSYS_CNTLR	The interface associated with this session is not the system controller.
-1073807271	VI_ERROR_RESP_PENDING	A previous response is still pending, causing a multiple query error.
-1073807275	VI_ERROR_NSUP_VAR_WIDTH	Cannot support source and destination widths that are different.
-1073807278	VI_ERROR_INV_WIDTH	Invalid access width specified.
-1073807301	VI_ERROR_QUEUE_ERROR	Unable to queue the asynchronous operation.
-1073807303	VI_ERROR_IN_PROGRESS	Unable to queue the asynchronous operation because there is already an operation in progress.
-1073807312	VI_ERROR_ABORT	User abort occurred during transfer.
-1073807313	VI_ERROR_NENABLED	You must be enabled for events of the specified type in order to receive them.
-1073807315	VI_ERROR_QUEUE_OVERFLOW	The event queue for the specified type has overflowed (usually due to previous events not having been closed).
-1073807327	VI_ERROR_INV_ACCESS_KEY	The access key to the specified resource is invalid.
-1073807328	VI_ERROR_INV_LOCK_TYPE	The specified type of lock is not supported by this resource.
-1073807333	VI_ERROR_INV_DEGREE	Specified degree is invalid.

Index

Numbers

3D graph controls for Windows, 2-9

A

accessing recently opened files, 2-4 to 2-5
ActiveX
    Automation Open function improvements, 2-14
    compatibility VIs for ActiveX functions, 1-16
    enhancements for Windows, 2-13 to 2-14
        ring enhancements, 2-13
        support for events, 2-13 to 2-14
        working with events, 2-13 to 2-14
    Event functions for Windows, 2-14
    IVI instrument drivers and Active X, 2-12 to 2-13
    manual additions, A-1 to A-2
Application Builder
    Run-Time Engine for, 2-46
    upgrading, 1-19
applications, building and distributing, 2-41 to 2-46
    building applications, 2-42 to 2-43
    building executable programs, 2-41 to 2-46
    creating installer for Windows, 2-45
    customizing application features, 2-43 to 2-44
    modifying VIs as part of the build, 2-45
    Run-Time Engine for Application Builder in Windows, 2-46

B

building applications. *See* applications, building and distributing.

C

CDE (Common Desktop Environment) Window Manager, configuring, 1-14 to 1-15
compatibility issues between versions 4.1 and 5.*x*, 1-6
compatibility VIs
    ActiveX functions, 1-16
    new server functionality, 1-16
Concurrent PowerMAX
    installation patches, 1-4
    installation requirements (table), 1-4
    installing LabVIEW, 1-9
    thread allocation, A-7 to A-8
Configuration File VIs, A-6
configuration requirements (table), 1-2 to 1-4
    HP-UX systems, 1-4
    Linux systems, 1-4
    Power Macintosh, 1-3
    Sun systems, 1-3
    UNIX systems, 1-3
    Windows operating systems
        all Windows versions, 1-2
        Windows 95/98, 1-2
        Windows NT, 1-2
configuring LabVIEW windows on UNIX, 1-14 to 1-15
    CDE (Common Desktop Environment) Window Manager, 1-14 to 1-15
    HP VUE Window Manager, 1-14
    Motif Window Manager, 1-14 to 1-15
    Tab Window Manager, 1-14
controls and indicators
    3D graph controls for Windows, 2-9
    changes and enhancements, 2-7 to 2-8
    dialog controls, 2-8
    labels, 2-7
    templates for VIs and controls, A-7

converting VIs, 1-18
CVI Functional Panel Converter, manual changes for, A-2

D

data acquisition (DAQ)
DAQ enhancements, 2-39 to 2-40
DAQ Solution Wizard, 2-39
new syntax element for nonsequentially scanned SCXI module channels, 2-39 to 2-40
support for NI-DAQ for Windows and Macintosh, 2-39
installation notes, 1-10 to 1-11
DataSocket VIs for Windows, 2-47
dialog box, menu, and window enhancements, 2-1 to 2-6
accessing recently opened files, 2-4 to 2-5
Macintosh Navigation Services, 2-6
saving VIs for previous version, 2-4
scaling front panel objects, 2-1 to 2-3
searching in LabVIEW, 2-5 to 2-6
dialog controls, 2-8
Dialog Listbox control, 2-8
Dialog Recessed Frame control, 2-8 to 2-9
discontinued media, 1-5
discontinued platform support, 1-5
distributing applications. *See* applications, building and distributing.
distribution of LabVIEW
discontinued media, 1-5
discontinued platform support, 1-5
documentation. *See* manual clarifications and additions.
dragging and dropping VI icons, A-5

E

Easy Text Report VI, 2-26, 2-33
enhancements. *See* features and enhancements.
errors
launch errors on UNIX (table), 1-13
VISA error codes (table), A-9 to A-11
examples
enhancements to examples and activities, 2-62
examples and solutions files, 1-12
executable programs, building, 2-41 to 2-46
execution system selection, A-6

F

features and enhancements
building and distributing applications, 2-41 to 2-46
DAQ enhancements, 2-39 to 2-40
dialog box, menu, and window enhancements, 2-1 to 2-6
accessing recently opened files, 2-4 to 2-5
Macintosh Navigation Services, 2-6
saving VIs for previous version, 2-4
scaling front panel objects, 2-1 to 2-3
searching in LabVIEW, 2-5 to 2-6
examples and activities, 2-62
networking enhancements, 2-47 to 2-62
VIs, functions, and controls, 2-7 to 2-38
3D graph controls for Windows, 2-9
ActiveX enhancements, 2-13 to 2-14
ActiveX event functions for Windows, 2-14
changes to controls and indicators, 2-7 to 2-9
generating reports in Windows, 2-25 to 2-28

HiQ and MATLAB functionality for Windows, 2-15 to 2-22
integration of mathematics and signal processing VIs, 2-22 to 2-24
integration of Picture Control VIs, 2-24 to 2-25
property and invoke nodes, 2-9
Report Generation VI descriptions, 2-28 to 2-38
ring enhancements, 2-11 to 2-13
sound VIs for Windows and Macintosh, 2-25
VI server properties in reserved VIs and runtime systems, 2-9 to 2-11
VISA enhancements, 2-40 to 2-41
File Manager tool, manual changes for, A-3 to A-5
files, recently opened, accessing, 2-4 to 2-5
Find All Instances dialog box, 2-5 to 2-6
Find dialog box, 2-5
front panel images for the Web
animated front panel image (.monitor URL), 2-60 to 2-62
determining which front panels are visible, 2-57
front panel image formats, 2-58
publishing, 2-56
retrieving static image of VI front panel, 2-56
static front panel image (.snap URL), 2-58 to 2-60
URLs for front panel images, 2-58
viewing animated version of front panel, 2-57
front panel objects, scaling, 2-1 to 2-3
defining minimum window size, 2-3
limitations (note), 2-1
maintaining window proportions, 2-3
rules for, 2-1 to 2-2
setting all objects to scale, 2-3
setting one object to scale, 2-2

G

general interface features. *See* interface features.
GPIB installation notes, 1-10 to 1-11

H

help. *See* information resources for LabVIEW.
Help menu, adding VIs to, A-7
HiQ and MATLAB, 2-15 to 2-22
choosing script server, 2-19 to 2-20
configuring data type of terminal, 2-20 to 2-21
creating HiQ scripts, 2-16 to 2-17
creating MATLAB scripts, 2-18 to 2-19
debugging scripts, 2-21
error codes (table), 2-22
importing or exporting scripts, 2-19
installing HiQ, 1-12
script node, 2-16
scrolling through scripts, 2-20
versions required, 2-15
HP VUE Window Manager, configuring, 1-14
HP-UX systems
installation requirements (table), 1-4
installing LabVIEW for HP-UX 10.*x*, 1-8
HTTP services. *See* Internet/HTTP services.

I

Icon Editor changes, A-6
Icon palette, manual changes for, A-2 to A-3
indicators. *See* controls and indicators.
information resources for LabVIEW, 1-10, 1-17
installing LabVIEW, 1-6 to 1-11. *See also* upgrading to LabVIEW 5.1.
data acquisition notes, 1-10 to 1-11
HiQ for Windows, 1-12
HP-UX 10.*x*, 1-8
LabVIEW RT, 1-6

Linux, 1-8 to 1-9
Macintosh, 1-6 to 1-7
more information about LabVIEW, 1-9
notes, 1-9 to 1-10
PowerMAX, 1-8
requirements
Concurrent PowerMAX, 1-4
HP-UX systems (table), 1-4
Power Macintosh (table), 1-3
Sun systems (table), 1-3
UNIX systems (table), 1-3
Windows operating systems (table), 1-2
UNIX, 1-7
VXI notes, 1-10 to 1-11
Windows, 1-6 to 1-7
instrumentation, manual additions for, A-2
interface features, A-2 to A-6
Configuration File VIs, A-6
dragging and dropping VI icons, A-5
execution system selection, A-6
File Manager tool, A-3 to A-5
Icon and Text palettes, A-2 to A-3
Icon Editor, A-6
Macintosh Open Transport support, A-6
new preferences options, A-6
offscreen updates default value, A-6
Print to RTF/HTML feature, A-5
Internet Developers Toolkit for G, upgrading, 1-19
Internet/HTTP services, 2-47 to 2-62
animated front panel image (.monitor URL), 2-60 to 2-62
configuring the Web Server, 2-55
determining which front panels are visible, 2-57
front panel image formats, 2-58
publishing front panel images on the Web, 2-56
retrieving static image of VI front panel, 2-56
running the Web Server, 2-55
static front panel image (.snap URL), 2-58 to 2-60
URLs for front panel images, 2-58
viewing animated version of front panel, 2-57
Web Server Browser Access dialog box, 2-49 to 2-51
Web Server Configuration dialog box, 2-48 to 2-49
Web Server in LabVIEW, 2-48
Web Server Visible VIs dialog box, 2-52 to 2-54
invoke nodes, 2-9
IVI instrument drivers and Active X, 2-12 to 2-13

L

labels for controls and indicators, 2-7
LabVIEW
about this addendum, *ix*
compatibility issues between versions 4.1 and 5.*x*, 1-16
examples and solutions, 1-12
information resources, 1-10, 1-17
upgrading to version 5.1, 1-17 to 1-19
LabVIEW RT, installing, 1-6
LabVIEW Test Executive, upgrading, 1-19
launch errors on UNIX (table), 1-13
Linux operating system, 1-4, 1-5, 1-8 to 1-9
low-level register I/O for Windows 95/98, 1-13

M

Macintosh Appearance Manager, 2-6
Macintosh computers. *See also* Power Macintosh.
installing LabVIEW, 1-7
data acquisition, VXI, and GPIB installation notes, 1-11

Power Macintosh installation requirements (table), 1-3
sound VIs, 2-25
support for NI-DAQ for Windows and Macintosh, 2-39
Macintosh Navigation Services, 2-6
Macintosh Open Transport support, A-6
manual clarifications and additions, A-1 to A-11
about this addendum, *ix*
ActiveX, A-1 to A-2
adding VIs to Project and Help menus, A-7
clarifications, A-8 to A-9
general interface features, A-2 to A-6
Configuration File VIs, A-6
dragging and dropping VI icons, A-5
execution system selection, A-6
File Manager tool, A-3 to A-5
Icon and Text palettes, A-2 to A-3
Icon Editor, A-6
Macintosh Open Transport support, A-6
new preferences options, A-6
offscreen updates default value, A-6
Print to RTF/HTML feature, A-5
instrumentation, A-2
multithreading, A-1
templates for VIs and controls, A-7
thread allocation on Concurrent PowerMAX and Solaris 2, A-7 to A-8
VISA error codes (table), A-9 to A-11
mathematics VIs
integration with signal processing VIs, 2-22 to 2-23
Mathematics palette, 2-23
Signal Processing palette, 2-24
MATLAB. *See* HiQ and MATLAB.
menu enhancements. *See* dialog box, menu, and window enhancements.
Motif Window Manager, configuring, 1-14 to 1-15
multithreading, manual additions for, A-1

N

networking enhancements, 2-47 to 2-62
DataSocket VIs for Windows, 2-47
Internet/HTTP services, 2-46 to 2-61
animated front panel image (.monitor URL), 2-60 to 2-62
configuring the Web Server, 2-55
determining which front panels are visible, 2-57
front panel image formats, 2-58
publishing front panel images on the Web, 2-56
retrieving static image of VI front panel, 2-56
running the Web Server, 2-55
static front panel image (.snap URL), 2-58 to 2-60
URLs for front panel images, 2-58
viewing animated version of front panel, 2-57
Web Server Browser Access dialog box, 2-49 to 2-51
Web Server Configuration dialog box, 2-48 to 2-49
Web Server in LabVIEW, 2-48
Web Server Visible VIs dialog box, 2-52 to 2-54
new features. *See* features and enhancements.
NI-DAQ for Windows and Macintosh, 2-39
nonsequentially scanned SCXI module channels, syntax element for, 2-39 to 2-40

O

offscreen updates default value, A-6
operating systems. *See also* specific operating system, e.g., UNIX operating system.

discontinued platform support, 1-5
installation requirements (table), 1-2 to 1-4
installing LabVIEW, 1-6 to 1-11
Save for Previous option (note), 1-5

P

Picture Control VIs, 2-24 to 2-25
Power Macintosh. *See also* Macintosh computers.
installation requirements (table), 1-3
PowerMAX operating system. *See* Concurrent PowerMAX.
preferences options, new, A-6
Print to RTF/HTML feature, A-5
Professional G Developers Toolkit, upgrading, 1-19
programs, building. *See* applications, building and distributing.
Project menu, adding VIs to, A-7
property nodes, 2-9

R

report generation, 2-25 to 2-28
capabilities of Report Generation VIs, 2-25 to 2-26
Easy Text Report VI overview, 2-26
hints, 2-27
Report in/Report out parameters, 2-27
tokens, 2-27 to 2-28
Report Generation palette, 2-28
Report Generation VIs, 2-28 to 2-39
Append File to Report, 2-34
Append Numeric Table to Report, 2-35
Append Report Text, 2-34
Append Text Table to Report, 2-35
capabilities, 2-25 to 2-26
Clear Report, 2-35
Clear Report Text, 2-35
Dispose Report, 2-36
Easy Text Report, 2-26, 2-33
Get Report Settings, 2-36
New Report, 2-36
New Report Line, 2-36
New Report Page, 2-37
parameter descriptions, 2-29 to 2-33
Print Report, 2-37
Set Report Font, 2-37
Set Report Footer Text, 2-37
Set Report Header Text, 2-38
Set Report Margins, 2-38
Set Report Orientation, 2-38
Set Report Tab Width, 2-39
required system configuration (table), 1-1 to 1-4
ring enhancements, 2-11 to 2-13
ActiveX, 2-13
IVI instrument drivers and ActiveX, 2-12 to 2-13
ring constant examples (figure), 2-11
runtime VIs, setting properties for, 2-9 to 2-10

S

saving for previous version
Save for Previous option (note), 1-5
saving VIs, 2-4
scaling front panel objects, 2-1 to 2-3
defining minimum window size, 2-3
limitations (note), 2-1
maintaining window proportions, 2-3
rules for, 2-1 to 2-2
setting all objects to scale, 2-3
setting one object to scale, 2-2
scripts. *See* HiQ and MATLAB.
SCXI module channels, nonsequentially scanned, syntax element for, 2-39 to 2-40
Search Results dialog box, 2-5 to 2-6
searching in LabVIEW, 2-5 to 2-6

Find All Instances and Search Results dialog box, 2-5 to 2-6
Find dialog box, 2-5
server functionality
compatibility VIs for, 1-16
VI server properties in reserved VIs and runtime systems, 2-9 to 2-11
Signal Generator by Duration VI, A-2
signal processing VIs
integration with mathematics VIs, 2-22 to 2-23
Mathematics palette, 2-23
Signal Processing palette, 2-24
Solaris 2
installing LabVIEW, 1-7 to 1-8
thread allocation, A-7 to A-8
solutions and examples, 1-12
sound VIs for Windows and Macintosh, 2-25
SPARCstation 5 systems, problems with, 1-15 to 1-16
Sun systems
data acquisition, VXI, and GPIB installation notes, 1-11
installation requirements (table), 1-3
problems with SPARCstation 5 systems, 1-15 to 1-16

T

Tab Window Manager, configuring, 1-14
templates for VIs and controls, A-7
Text palette, manual changes for, A-2 to A-3
threads
multithreading, A-1
thread allocation on Concurrent PowerMAX and Solaris 2, A-7 to A-8
3D graph controls for Windows, 2-9
tokens
descriptions (table), 2-27 to 2-28
report generation, 2-27 to 2-28
toolkits, upgrading, 1-18 to 1-19

U

UNIX operating system
configuring LabVIEW windows, 1-14 to 1-15
CDE (Common Desktop Environment) Window Manager, 1-14 to 1-15
HP VUE Window Manager, 1-14
Motif Window Manager, 1-14 to 1-15
Tab Window Manager, 1-14
installation requirements (table), 1-3
installing LabVIEW, 1-7 to 1-9
HP-UX 10.*x*, 1-8
Linux, 1-8 to 1-9
PowerMAX, 1-9
Solaris 2, 1-7
launch errors on UNIX (table), 1-13
upgrading to LabVIEW 5.1, 1-17 to 1-19
application builder libraries and toolkits, 1-18 to 1-19
converting VIs, 1-18

V

VI icons, dragging and dropping, A-5
VIs. *See also* Report Generation VIs.
adding VIs to Project and Help menus, A-7
compatibility VIs
ActiveX functions, 1-16
new server functionality, 1-16
Configuration File VIs, A-6
converting, 1-18
DataSocket VIs for Windows, 2-47
integration
mathematics and signal processing VIs, 2-22 to 2-24
Picture Control VIs, 2-24 to 2-25
runtime VIs, setting properties for, 2-9 to 2-10

saving for previous version, 2-4
Signal Generator by Duration VI, A-2
sound VIs for Windows and Macintosh, 2-25
templates for VIs and controls, A-7
VI server properties in reserved VIs and runtime systems, 2-9 to 2-11
VISA enhancements, 2-40 to 2-41
VISA error codes (table), A-9 to A-11
VISA GPIB Control REN function, 2-41
VISA VXI Cmd or Query, 2-41
VXI installation notes, 1-10 to 1-11

W

Web Server. *See also* Internet/HTTP services.
configuring, 2-55
overview, 2-48
running, 2-55
Web Server Browser Access dialog box, 2-49 to 2-51
example TCIP/IP access entries (table), 2-51
illustration, 2-49
options (table), 2-50
Web Server Configuration dialog box, 2-48
Web Server Visible VIs dialog box, 2-52 to 2-55
examples of Visible VIs list entries (table), 2-54
illustration, 2-52
options (table), 2-53
wildcard characters in Visible VIs list (table), 2-54
window enhancements. *See* dialog box, menu, and window enhancements.
window managers. *See* configuring LabVIEW windows on UNIX.
windows for front panel
defining minimum size, 2-3
maintaining proportions with monitor resolution, 2-3
Windows operating systems
ActiveX enhancements, 2-13 to 2-14
Automation Open function, 2-14
Event functions, 2-14
ring enhancements, 2-13
support for ActiveX events, 2-13
HiQ and MATLAB, 2-15 to 2-22
choosing script server, 2-19
configuring data type of terminal, 2-20
creating HiQ scripts, 2-16 to 2-17
creating MATLAB scripts, 2-18 to 2-19
debugging scripts, 2-21
error codes (table), 2-22
importing or exporting scripts, 2-19
scrolling through scripts, 2-20
installation requirements (table)
all Windows versions, 1-2
Windows 95/98, 1-2
Windows NT, 1-2
installing LabVIEW
data acquisition, VXI, and GPIB installation notes, 1-10 to 1-11
procedure for, 1-6 to 1-7
low-level register I/O for Windows 95/98, 1-13
report generation, 2-25 to 2-28
Run-Time Engine for Application Builder, 2-41 to 2-46
sound VIs, 2-25
support for NI-DAQ for Windows and Macintosh, 2-39
World Wide Web. *See* Internet/HTTP services.